MASTER EN SCIENCES INFORMATIQUES

Acquisition de spécifications à partir du langage naturel

Thiran, Philippe

Award date:
1997

Awarding institution:
Universite de Namur

Facultés Universitaires Notre-Dame de la Paix
Institut d'Informatique
Namur

Acquisition de spécifications à partir du langage naturel

Philippe Thiran

Mémoire présenté en vue
de l'obtention du grade de
Licencié et Maître en Informatique

Promoteur : J.-L. Hainaut

Année académique 1996-1997

Résumé

Ce mémoire présente une méthode de construction d'un schéma conceptuel de données à partir d'un énoncé en **langage naturel**. Nous caractérisons notre **méthode** de conception par trois composants de base : une démarche fondée sur des modèles et mise en œuvre par des outils logiciels. Notre méthode utilise deux **modèles** conceptuels : le modèle Entité-Association et le modèle sémantique élémentaire qui permet une représentation immédiate des énoncés exprimés en langage naturel. La **démarche** est formée d'étapes qui assurent la guidance et le contrôle de la construction du schéma conceptuel à partir d'un texte. Elle repose sur une approche linguistique. Celle-ci est fondée sur une formalisation des mécanismes linguistiques par lesquels l'analyste est capable d'abstraire la réalité perçue en terme de concepts. Cette approche linguistique est mise en œuvre par des **outils logiciels**. Ils supportent notre démarche et visent à automatiser le processus par lequel un analyste produit un schéma conceptuel à partir d'un énoncé.

Abstract

In this work, we present a method to build a data conceptual schema from informations verbalized in natural language. This method has three components : a process, based on models, and realized by softwares. We use two conceptual models : the Entity-Relationship model and the « elementary semantic model », which allows us to represent texts expressed in natural language. The process is divided into steps guiding and controlling the construction of conceptual schemas from texts. It is based on a linguistic approach obtained by formalizing the linguistic mechanisms used by analysts to translate the Universe of Discourse into concepts. This linguistic approach is realized by softwares aimed to mimic the process used by an analyst to produce a conceptual schema form texts.

Avant-propos

Ce mémoire n'aurait certainement pas vu le jour sans l'aide de plusieurs personnes. Je tiens ici à les remercier.

Mes premiers remerciements vont d'abord à mon directeur de mémoire, Monsieur le professeur Jean-Luc Hainaut. Je lui suis particulièrement reconnaissant pour ses précieux conseils, sa constante disponibilité et l'intérêt qu'il a porté à ce travail.

Je remercie également Monsieur Patrick Heymans pour son intervention dans certains aspects spécifiques de ce travail.

Ma pensée se porte également vers Monsieur Arnaud Deflorenne, Monsieur Vincent Englebert et toute l'équipe DB-MAIN pour leur disponibilité et leur aide.

Enfin, j'aimerais remercier mes parents qui n'ont jamais cessé de me soutenir et de m'aider durant ces années universitaires.

S'il se trouve des personnes que je n'ai pas citées et qui ont pu contribuer directement ou indirectement à la réalisation de ce mémoire, qu'elles en soient également remerciées.

Table des matières

1. Introduction

La conception d'une base de données est difficile. Il n'existe pas de démarche simple pouvant générer à partir des besoins en information une structure de base de données. Le choix des objets à représenter, la description de la structure de ces objets et la définition des contraintes qui régissent la manipulation dépendent essentiellement de la perception que l'analyste se fait de la réalité du système d'information. Toutefois, même si l'automatisation du processus de conception ne peut être entreprise, il est possible de fournir des **aides** tant dans la représentation des données que dans la démarche à suivre pour arriver à un schéma conceptuel cohérent ([BODART, 94]). Trois types d'aides sont généralement fournies à l'analyste :

1. des **modèles** permettant de décrire les informations à différents niveaux d'abstraction allant de la représentation conceptuelle à la représentation physique ;
2. une **démarche** décrivant un savoir-faire pratique en termes d'étapes (étude d'opportunité, analyse conceptuelle, conception technique, réalisation et implantation, par exemple) de procédures (comment réaliser une étape donnée), de vérification et de validation (par confrontation des spécifications ou par prototypage) ;
3. des **outils informatiques** aidant à la production de rapports de conception (éditeurs graphiques ou de textes, rapports documentaires, par exemple), à la réalisation d'une étape (générateur de schémas, générateur de codes, etc.), à la conduite de démarches (gestionnaires de projets, par exemple), à la validation (prototypes, maquettes, outils de simulation, générateur de tests, par exemple), à la maintenance de bases de données et à la rétro-ingénierie (outils de transformation de schémas, assistants pour la manipulation de schémas, outils d'analyse des programmes sources, par exemple).

Ces trois classes d'aide sont fournies par la plupart des **méthodes** utilisées aujourd'hui. Il existe actuellement plusieurs méthodes plus ou moins complexes. Nous citerons, à titre d'illustration, les méthodes NIAM ([VERHEIJEN, 82]), MERISE ([TARDIEU, 86]), REMORA ([ROLLAND, 88]), IDA ([BODART, 94]) et CSDP ([HALPIN, 95]).

L'approche méthodologique généralement utilisée pour concevoir une base de données est résumée par le schéma de la figure 1.1. Ce schéma est une simplification de l'approche par niveaux proposée dans [HAINAUT, 94].

L'analyse conceptuelle consiste à élaborer une description complète du système d'information qui soit indépendante de toute technique. C'est la formalisation du réel perçu. Elle est constituée, entre autres, d'un schéma conceptuel des données.

La **conception logique** consiste à traduire le schéma conceptuel en un schéma logique. On transforme les concepts spécifiés au niveau conceptuel dans le langage supporté par le système de gestion de bases de données (SGBD) qu'on va utiliser.

La **conception physique** définit la structure du stockage physique des données et des méthodes d'accès à ces données. La solution physique doit répondre aux trois propriétés suivantes : elle doit être correcte par rapport à l'analyse conceptuelle, efficace et exécutable.

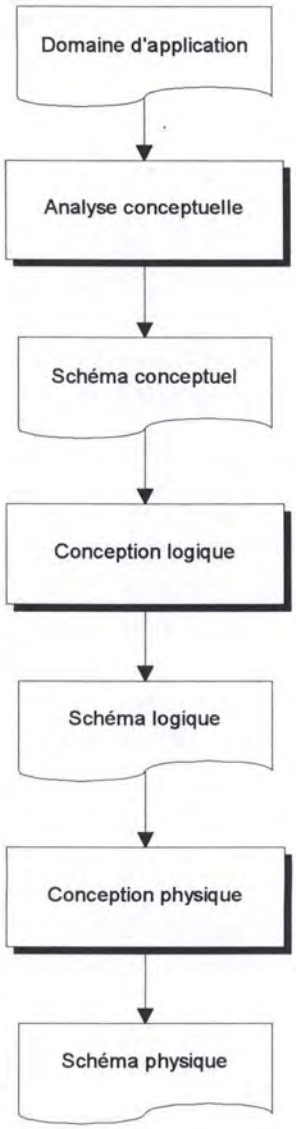


Figure 1.1. Conception d'une base de données

Dans notre mémoire, nous nous intéressons uniquement à la première phase de la conception, l'analyse conceptuelle et plus précisément à l'**analyse conceptuelle des données**.

1.1 Analyse conceptuelle

L'analyse conceptuelle est à la fois **cruciale** car elle conditionne la qualité de la base de données et **complexe** car elle consiste en une activité de modélisation ([BATINI, 92], [HAINAUT, 94]). L'analyse conceptuelle de données est présentée sous la forme d'un **schéma conceptuel**. Un schéma conceptuel est une représentation de la structure des informations indépendamment de la manière dont celles-ci sont stockées physiquement.

Pour remplir totalement son rôle, il est souhaitable qu'un schéma conceptuel soit ([BODART, 94]) :

1. **communicable** : précis, simple et standard ;
2. **cohérent** : sans contradiction, ni ambiguïté ;
3. **conforme et complet** par rapport à un modèle de référence ;
4. **fidèle** : représentation sans biais et sans déformation ;
5. **non redondant** : ne tenant compte que les éléments qui sont strictement nécessaires.

Les sources d'information utilisées lors de l'élaboration d'un schéma conceptuel peuvent être extrêmement diverses. Dans ce mémoire, nous nous limitons à un seul type d'information constitué d'un énoncé en **langage naturel**. C'est en effet par entretiens, d'une part et par lecture de documents écrits, d'autre part, que s'opère essentiellement le transfert de connaissance liée au domaine d'application ([ROLLAND, 91]).

La construction d'un schéma conceptuel à partir d'un texte correspond à une démarche de modélisation basée sur une **approche linguistique.** Cette démarche qui trouve ses origines dans les travaux de Senko ([SENKO, 73]) a fait l'objet de nombreuses recherches. Parmi celles-ci, citons la méthode NIAM créée par Nijssen ([VERHEIJEN, 82]), la méthode REMORA de Rolland ([ROLLAND, 88]) et la méthode CSDP de Halpin ([HALPIN, 96]).

1.2 Revue de la littérature

Le processus de conception d'un schéma conceptuel a successivement conduit les chercheurs à développer des modèles, des méthodes de conception et des outils logiciels facilitant et renforçant l'usage des méthodes.

1.2.1 Modèles

Il existe trois grandes classes de modèles ; elles sont présentées dans l'ordre chronologique de leur émergence.

Le **modèle relationnel** ([CODD, 70]) a été précurseur. La théorie relationnelle est aujourd'hui largement utilisée dans le monde des bases de données. Toutefois les concepts relationnels sont trop limités pour pouvoir représenter complètement la sémantique du domaine d'application ([DELOBEL, 91]).
D'autres travaux ont abouti à la définition **de modèles sémantiques**. L'objectif de ces modèles est de représenter le plus possible de concepts sur un même schéma. Ces modèles présentent le monde réel comme une collection d'entités (parfois appelés objets) et d'associations entre ces entités. Ils permettent aussi la définition de contraintes décrivant les aspects statique, dynamique ou même temporel. Parmi ces modèles, nous pouvons distinguer deux classes : les modèles Entité-Association et les modèles binaires.

Les principaux concepts des **modèles Entité-Association** sont l'entité, l'association, les attributs (propriétés) et les cardinalités exprimant des contraintes sur les associations. Le modèle Entité-Association est actuellement le modèle conceptuel le plus populaire. C'est un article de Chen ([CHEN, 76]) qui est considéré comme l'expression de référence de l'approche Entité-Association.

Le **modèle binaire**, défini dans [ABRIAL, 74], est basé sur deux concepts : le concept de catégorie, qui permet de classifier les objets en types, et le concept de relation binaire entre deux catégories. Le modèle binaire est un modèle simple et bien adapté à la représentation immédiate des énoncés en langage naturel. En particulier, il permet de ne pas faire de distinction entre entités et propriétés de ces entités au sens classique du modèle Entité-Association. Le modèle NIAM ([VERHEIJEN, 82]) est un exemple de modèle binaire.

Les modèles sémantiques ne répondent qu'à une partie des problèmes de modélisation que rencontre l'analyste. Ils limitent la représentation du monde réel à des aspects statiques ou structurels. Il sont inaptes à prendre en compte les aspects dynamiques relatifs à l'évolution des données dans le temps.

Finalement, les **modèles dynamiques** qui sont les plus récents, tendent à intégrer la représentation de la structure des données et celle du comportement des données. Ces modèles sont généralement une extension des modèles sémantiques à des concepts (événement, action) permettant de représenter le comportement du système réel. Nous pouvons citer comme exemple le modèle dynamique de la méthode REMORA ([ROLLAND, 88]).

1.2.2 Méthodes de conception

Les méthodes de conception sont centrées sur l'utilisation des modèles conceptuels. La conception s'assimile dès lors à une activité de **modélisation** ; le résultat en est une représentation abstraite (schéma conceptuel) du monde réel construite à l'aide des concepts du modèle utilisé. Le processus de conception est généralement organisé en deux grandes étapes :

La **première étape** est une étape d'**analyse** et de **description**. Elle consiste d'abord à décrire le domaine d'application par un texte structuré. Ensuite, elle analyse les objets réels et regroupe ces objets en classe. Elle se termine par l'expression de ces objets à l'aide de concepts du modèle utilisé : modèle NIAM pour la méthode NIAM ([VERHEIJEN, 82]), modèle dynamique pour la méthode REMORA ([ROLLAND, 88]), modèle Entité-Association pour la méthode proposée par [BODART, 94] et modèle ORM pour la méthode CDSP ([HALPIN, 96]).

Ainsi l'analyste utilisant la méthode proposée par [BODART, 94] procédera de la façon suivante :

1. **Description du domaine d'application par des phrases.** Par exemple « `John lives in Paris` ».
2. **Analyse et généralisation** des phrases de manière à faire apparaître des regroupements d'objets réels. Par exemple, la phrase précédente est généralisée de la façon suivante :

« `The person 'John' lives in the town 'Paris'` », faisant apparaître les phénomènes `personne`, `ville` et `habite`.

3. **Expression** de ces phénomènes à l'aide de concepts du modèle Entité-Association[1]. Dans le cas précédent, l'analyste reconnaît deux types d'entité `Person` et `Town`, deux propriétés (attributs) `Person-Name` et `City-Name` ainsi que le type d'association `Living` qui associe les deux types d'entité.

Finalement, il aboutit au schéma Entité-Association de la figure 1.2.

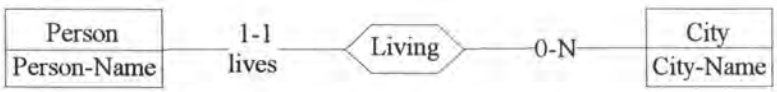


Figure 1.2. Schéma Entité-Association résultant de l'analyse

La **seconde étape** consiste en une activité de **normalisation**. Cette activité consiste à définir une « bonne représentation » des classes de phénomènes inventoriées dans la première étape. Les critères de bonne représentation sont dépendants des méthodes utilisées. Ce peut être l'élimination de la redondance, la complétude, etc. Chaque méthode traduit ses critères en termes de normes que l'analyste doit respecter dans l'élaboration du schéma conceptuel.

1.2.3 Outils logiciels d'aide à la conception de schéma conceptuel

L'environnement logiciel d'aide à la **conception de schéma conceptuel** tend à fournir un ensemble d'outils **passifs** (supports à la gestion de la conception) et **actifs** (supports à la conception en elle-même).

1.2.3.1 Outils passifs

Ces outils sont orientés vers l'assistance à la conception d'un schéma conceptuel. Ils apportent essentiellement des facilités de production de la documentation, à la mise en forme des résultats et au contrôle de la tâche de conception. Ces outils remplissent, plus ou moins complètement, trois fonctions :

1. **Gestion** des spécifications du schéma conceptuel à l'aide d'**éditeurs graphiques** permettant leur saisie, leur mémorisation, leur présentation graphique et leur manipulation.
2. **Validation** et **contrôle** du schéma conceptuel à l'aide d'**outils de simulation,** d'**analyseurs de schéma** ou encore de **langages d'interrogation** permettant à l'analyste d'explorer le schéma conceptuel.
3. **Génération automatique de documentation** à l'aide de générateur de texte au départ d'un schéma conceptuel. Aux Facultés Universitaires Notre-Dame de la Paix, dans le cadre du projet DB-MAIN, [DEFLORENNE, 96] a développé NATURAL, un programme

[1] Les concepts du modèle Entité-Association sont définis dans le chapitre 2.

permettant de générer deux types de texte au départ d'un schéma conceptuel réalisé dans l'atelier DB-MAIN. Il fournit soit un texte continu en langage naturel soit un ensemble de propositions à valider. Le texte continu est une description du schéma conceptuel. Le second document reprend tous les faits élémentaires et permet leur validation grâce à la présence de cases à cocher.

De manière générale, nous pouvons dire que les outils passifs n'apportent pas une aide suffisante dans la tâche de conception parce qu'ils se limitent à un rôle de documentation et de renseignement sur le résultat du processus de conception.

1.2.3.2 Outils actifs

Ces outils visent à aider activement l'analyste dans le processus créatif de production d'un schéma conceptuel. Ils souhaitent apporter une aide lors de la tâche de spécification. Ils interviennent sur le processus de conception lui-même, alors que les outils passifs aident à gérer le résultat du processus de conception.

L'idée initiale, ayant conduit à la réalisation de ces systèmes, est la construction d'outils permettant d'obtenir le schéma conceptuel de la manière la plus automatique possible. Pour ce faire ces outils basent leur aide sur la connaissance des règles de conception mais aussi sur la connaissance d'heuristiques de conception de schéma conceptuel.

Trois outils sont caractéristiques de cette approche : ACME réalisé par Kersten ([KERSTEN, 87]), OICSI développé par Proix ([PROIX, 89], [ROLLAND, 91], [ROLLAND, 92]) et KHEOPS développé dans le laboratoire PRiSM de l'université de Versailles ([AMBROSIO, 95]).

Cette classe d'outils est basée sur deux principes essentiels : l'utilisation d'une **approche linguistique** et l'utilisation des **techniques d'Intelligence Artificielle** pour la représentation des connaissances. Ces outils aident l'analyste à produire un schéma conceptuel à partir d'un ensemble d'énoncés descriptifs du domaine. Le processus de construction est organisé en trois étapes :

1. la première étape consiste à saisir la description du domaine d'application énoncée sous forme de phrases en français pour les outils KHEOPS et OICSI et en anglais pour l'outil ACME.
2. à les analyser, à les traduire sous forme d'arbres syntaxiques et enfin à les interpréter pour aboutir à une première version du schéma conceptuel (un schéma dynamique pour OICSI et un schéma Entité-Association pour KHEOPS et ACME);
3. la troisième étape a pour but de transformer tout en l'enrichissant, le schéma conceptuel initial. Cette transformation est menée interactivement avec l'analyste à qui l'outil propose des choix.

1.3 Objectif du mémoire

Le mémoire présente une **méthode** de conception d'un schéma conceptuel des données à partir d'un texte en anglais. L'anglais a été préféré au français pour sa simplicité grammaticale et lexicale. Notre méthode propose une **démarche** fondée sur des **modèles** et mise en œuvre à l'aide d'**outils logiciels** (cfr. figure 1.3).

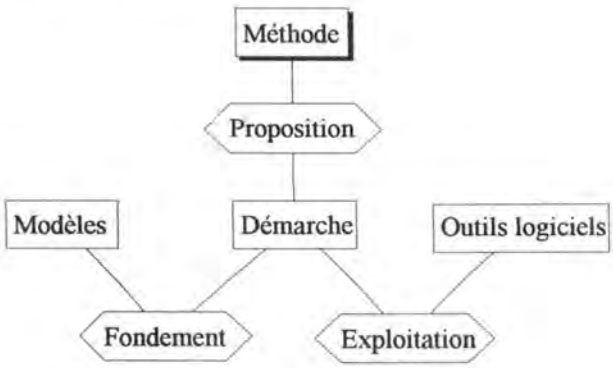


Figure 1.3. Méthode

1.3.1 Modèles

Il existe de nombreux modèles conceptuels de structuration des informations. Nous renvoyons le lecteur à [OLLE, 82] pour un exposé de ceux-ci. Parmi ceux-ci, nous avons retenu le modèle **Entité-Association standard** et le modèle **binaire**.

Le modèle **Entité-Association standard** (EA) est actuellement un des modèles conceptuels le plus utilisé dans l'analyse et la modélisation des systèmes d'information. Il permet une représentation des informations indépendamment de la manière dont celles-ci sont stockées physiquement. Ce modèle rencontre une forte audience parmi les spécialistes des systèmes d'information et des bases de données : ils lui attribuent de grandes qualités de communication et une bonne capacité de représentation des informations appartenant au réel perçu ([BODART, 94]).

Dans notre méthode, nous nous limitons à une variante simplifiée du modèle Entité-Association. Nous ferons référence à ce modèle EA simplifié par le vocable : **modèle EA de base**. Quoique limitée, la puissance d'expression de ce modèle est suffisante pour modéliser la plupart des situations exprimables en langage naturel.

Le **modèle sémantique élémentaire** est basé sur le modèle binaire NIAM ([NIJSSEN, 89]). Le modèle sémantique élémentaire a été introduit dans ce mémoire car il est simple et adapté à la représentation immédiate des énoncés exprimés en langage naturel. Il est un intermédiaire entre le langage naturel et le schéma EA de base.

1.3.2 Démarche

La démarche proposée est constituée d'un ensemble de règles qui assurent la guidance et le contrôle du processus de conception. Elle met l'accent sur la façon de conduire et de dérouler le processus de conception. Elle est basée sur l'**approche linguistique** proposée par Nijssen ([VERHEIJEN, 82]). Le processus s'exprime alors comme suit : à partir d'un énoncé exprimé en langage naturel, construire un schéma EA de base représentant les concepts et les faits exprimés dans l'énoncé.

1.3.3 Outils logiciels

Les deux outils logiciels NATURAL EDITOR et NATURAL DB-MAIN que nous avons développés sont une **aide active** au processus de conception. NATURAL EDITOR met à la disposition de l'analyste une interface en **langage naturel** qui lui permet d'exprimer le domaine d'application sous forme d'un texte. NATURAL DB-MAIN construit automatiquement le schéma sémantique élémentaire. Il est intégré à l'atelier DB-MAIN ([HAINAUT, 96]) qui assure, en particulier, la présentation graphique des schémas conceptuels et la transformation du schéma sémantique élémentaire en un schéma Entité-Association de base.

1.4 Organisation du mémoire

Le **présent chapitre** a déjà présenté les objectifs poursuivis et a établi un état de la question afin de placer notre méthode dans le courant de la recherche.

Le **chapitre deux** spécifie les modèles sur lesquels est fondée notre démarche. En premier lieu, il définit le modèle EA de base et le modèle sémantique élémentaire. Il présente ensuite un certain nombre de mécanismes de transformation qui permettent de passer d'un schéma à l'autre.

Le **chapitre trois** propose une démarche formée d'étapes et de règles en vue de maîtriser les étapes de la conception d'un schéma conceptuel à partir d'un énoncé exprimé en langage naturel.

Le **chapitre quatre** présente l'approche linguistique sur laquelle est basée notre démarche. Nous tentons de comprendre les mécanismes linguistiques utilisés par un analyste qui, à partir d'un texte, construit un schéma conceptuel. Nous exposons ensuite les fondements de l'approche linguistique basée sur la théorie des cas de Fillmore.

Le **chapitre cinq** montre, sans tenir compte de la technique, de quelle manière nous avons mis en œuvre l'approche linguistique : il spécifie comment implanter un processus qui génère automatiquement un schéma sémantique élémentaire à partir d'un texte simple.

Le **chapitre six** présente les outils logiciels que nous avons développés et qui supportent notre démarche.

Le **chapitre sept** propose finalement d'appliquer notre méthode à une étude de cas. Nous présentons les différentes étapes de notre démarche ainsi que les outils logiciels qui la supportent.

2. Modèles

2.1 Introduction

Dans ce chapitre, nous exposons les différents **modèles** conceptuels que nous utilisons dans le cadre de la démarche méthodologique que nous proposons (cfr. figure 2.1).

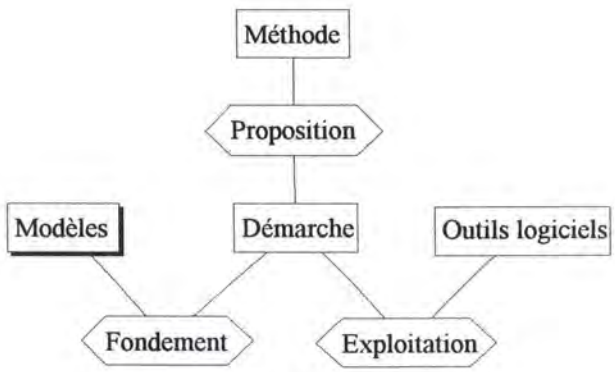


Figure 2.1. Méthode et modèles

2.2 Modèle Entité-Association de base

Comme nous l'avons stipulé dans l'introduction, le modèle EA de base est un sous-ensemble des formalismes utilisés dans le modèle EA standard. Dans ce paragraphe, nous définissons les concepts du modèle EA de base en précisant ses limitations par rapport au modèle EA standard. Nous adaptons une approche inspirée de [HAINAUT, 94], [BODART, 94] : un modèle permettant d'exprimer la sémantique des données mémorisables et/ou véhiculables à l'aide des concepts d'entité et d'attribut.

2.2.1 Entité

Une **entité** est un objet concret ou abstrait appartenant au réel perçu à propos duquel nous voulons enregistrer des informations. Cependant, dans un processus de description, nous ne nous intéressons pas à chaque objet individuel mais nous envisageons les classes d'objets, ou **type d'entité (TE)**. Dans le contexte d'une bibliothèque, nous pouvons repérer comme objets concrets : des livres, des auteurs et des emprunteurs. Dans notre exemple, nous définissons naturellement trois types d'entité : `Book, Subscriber, Author.`

Les types d'entité sont représentés graphiquement comme indiqué à la figure 2.2.

Figure 2.2. Représentation graphique des types d'entité

2.2.2 Attribut

Un **attribut** est une caractéristique ou qualité d'une entité. Ainsi chaque emprunteur est caractérisé par un numéro d'emprunteur, un nom, une adresse et un numéro de téléphone. Graphiquement, le nom d'un attribut est inscrit dans la partie inférieure du rectangle qui représente le TE (cfr. figure 2.3).

Figure 2.3. Représentation graphique d'un attribut

2.2.2.1 Attribut facultatif

Il est possible que, pour un type d'entité donné, la valeur d'un attribut ne soit pas connue ou que cette valeur n'ait pas de sens. Si nous admettons qu'un attribut puisse ne pas avoir de valeur pour certaines entités, nous le déclarerons **facultatif**. Sinon, l'attribut est **obligatoire**.

Graphiquement, un attribut facultatif est indiqué par la notation `[0-1]` qui signifie que, pour un type d'entité donné, l'attribut peut prendre 0 ou 1 valeur. Dans notre exemple, supposons que le numéro de téléphone de l'emprunteur n'est pas nécessairement connu : l'attribut `Phone Number` est facultatif. Le type d'entité `Subscriber` se présente maintenant comme illustré à la figure 2.4.

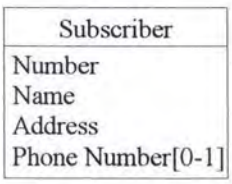


Figure 2.4. Représentation graphique d'un attribut facultatif

2.2.2.2 Attribut multivalué

Un attribut est **multivalué** (ou répétitif) si pour une entité ou une association, il peut prendre plusieurs valeurs d'un même type. Un tel attribut est caractérisé par un couple de valeurs `[i-j]` où `i` représente le nombre minimum et `j` le nombre maximum de valeurs que cet attribut peut prendre pour une entité donnée. Les nombres formant le couple de valeurs `i-j` sont respectivement appelés **cardinalité minimale** (`i`) et **cardinalité maximale** (`j`). Toute valeur non négative est admise pour `i` et `j` à condition que $i \leq j$ et que $j \geq 1$. Si nous désirons, dans notre exemple, enregistrer cinq numéros de téléphone, le type d'entité `Subscriber` est alors représenté comme indiqué dans la figure 2.5.

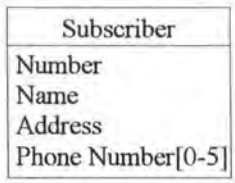


Figure 2.5. Représentation graphique d'un attribut multivalué

2.2.3 Attribut décomposable

Un attribut est **décomposable** si chacune de ses valeurs est constituée d'un assemblage de valeurs significatives plus petites. L'attribut `Address` de `Subscriber` peut être constitué d'attributs plus élémentaires `Number`, `Street`, `Zip Code`, et `City Name`. Graphiquement, les attributs décomposables sont représentés de la façon suivante :

Figure 2.6. Représentation graphique d'un attribut décomposable

Un attribut qui n'est pas décomposable est un attribut **atomique**. Dans le modèle sémantique élémentaire, nous ne considérons que les attributs atomiques.

2.2.4 Association

Souvent, les différentes entités du domaine d'application sont liées entre elles par des **associations**. Dans notre exemple, un emprunteur peut emprunter des livres : il y a donc une association entre l'emprunteur et le livre qu'il a emprunté.

Les associations vérifiant les mêmes propriétés constituent un **type d'association (TA)**. Ainsi, notre exemple présente un type d'association (appelé `Borrow`) entre les types d'entité `Subscriber` et `Book`. De même, en admettant qu'un livre est écrit par un nombre quelconque d'auteurs, nous définissons un type d'association `Written` entre les types d'entité `Book` et `Author`.

Un type d'association est représenté par un hexagone relié par des segments de droite aux rectangles qui représentent les TE sur lesquels est défini le TA (cfr. figure 2.7).

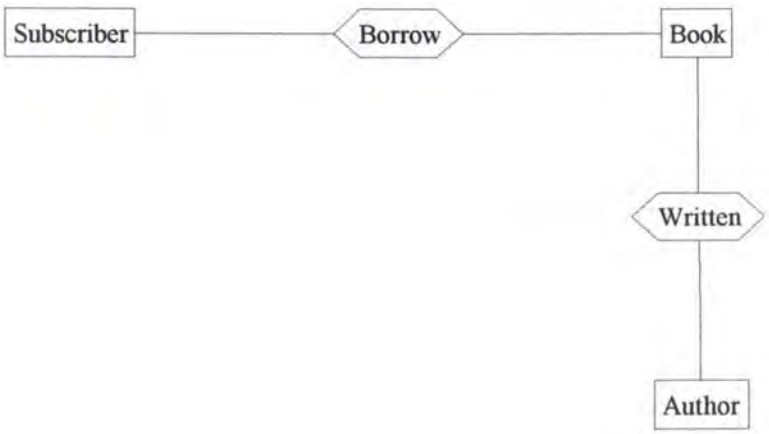


Figure 2.7. Représentation graphique d'un type d'association

2.2.4.1 Degré d'un type d'association

Un type d'association peut relier un nombre quelconque de types d'entité. Le nombre de types d'entité, non nécessairement distinct, sur lesquels le type d'association est défini est appelé **degré du type d'association.** Dans le modèle sémantique élémentaire, nous n'admettons que les types d'association binaire (de degré 2).

2.2.4.2 Rôle

Chaque type d'entité TE participant à un type d'association joue un rôle dans le cadre de ce type d'association. Graphiquement, le nom du rôle est indiqué sous le segment qui relie le TE au TA (cfr. figure 2.8).

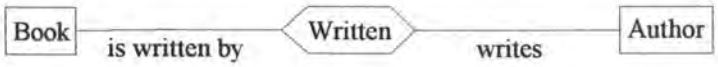


Figure 2.8. Représentation graphique d'un rôle

Dans notre exemple, le rôle joué par `Book` dans le cadre du TA `Written` est `is written by` tandis que celui joué par `Author` est `writes`. Le schéma peut alors se traduire facilement en langage naturel : `A book is written by an author` et `An author writes several books`.

2.2.4.3 Cardinalité d'un type d'association

Soit R, un type d'association défini sur les types d'entité TE_1, ..., TE_n. La cardinalité d'un type d'association R est définie par un ensemble de couples d'entiers (`i`,`j`) où :

- `i` indique le nombre **minimum** d'associations auxquelles une entité de TE_i doit participer ;
- `j` indique le nombre **maximum** d'associations pour toute entité de TE_i.

Les nombres formant le couple de valeurs `i-j` sont respectivement appelés **cardinalité minimale** (`i`) **et cardinalité maximale** (`j`). Toute valeur non négative est admise pour `i` et `j` à condition que $i \leq j$ et que $j \geq 1$. Pour une cardinalité maximale aussi grande que l'on veut, on affecte la variable `N`.

Dans notre exemple, un emprunteur peut emprunter plusieurs livres (un nombre indéfini) ou alors n'en emprunter aucun (nous acceptons les emprunteurs « potentiels »). Les cardinalités du type d'entité `Subscriber` dans le cadre du type d'association `Borrow` sont indiquées par `0-N` : `A subscriber can borrow several books`. En outre, un livre ne peut être emprunté que par un et un seul emprunteur : les cardinalités du type d'entité `Book` dans le cadre du type d'association `Borrow` sont `1-1` : `A book must be borrowed by one and only one subscriber.`

Graphiquement, la cardinalité est indiquée au-dessus du nom du rôle (cfr. figure 2.9).

Figure 2.9. Représentation graphique des cardinalités

2.2.4.4 Type d'association cyclique

Le **type d'association cyclique** établit une correspondance entre un type d'entité et lui-même. Par exemple (cfr. figure 2.10), considérons le cas d'une relation hiérarchique entre personnes : une personne peut assumer les deux rôles suivants : supérieur ou subordonné. Dans ce cas, il est nécessaire de nommer le rôle joué pour distinguer chaque entité dans une association.

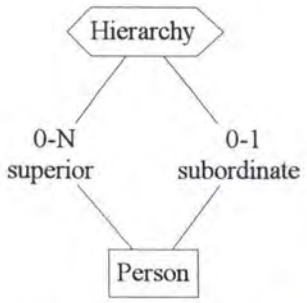


Figure 2.10. Représentation graphique d'un type d'association récursif

2.2.5 Identifiant

Chaque type d'entité peut posséder un (ou plusieurs) **identifiant** qui permet de repérer univoquement chaque entité de ce type. Par exemple, les entités du type `Subscriber` sont identifiées par leur numéro (`Number`) unique. Cela signifie qu'à tout instant, il n'existe pas deux entités `Subscriber` qui possèdent la même valeur de `Number`.

Le cas du type d'entité `Book` est plus complexe. Nous considérons, en effet, qu'il n'existe pas deux livres possédant le même titre et écrit par le même auteur. Le type d'entité `Book` est donc identifié par l'attribut `Title` et le type d'entité `Author`.

En toute généralité, un identifiant d'un type d'entité peut être constitué :

1. de un ou plusieurs attributs ;
2. au moins un rôle assumé par le type d'entité ;
3. d'un groupe formé par un ou plusieurs de ses attributs et par un ou plusieurs rôles.

Les constituants de l'identifiant d'un type d'entité TE sont indiqués graphiquement sous le rectangle représentant le `TE` avec la notation « `id : ...` ». Lorsqu'un identifiant est constitué d'un type d'entité TE_i relié à `TE` via le type d'association `TA`, nous indiquons ce constituant de l'identifiant par la notation $TA.TE_i$. Si l'identifiant de `TE` est constitué uniquement d'attributs de `TE`, alors ces attributs sont soulignés (cfr. figure 2.11).

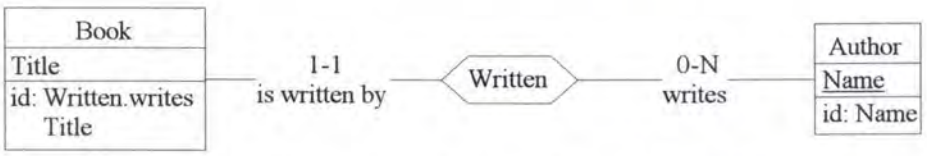


Figure 2.11. Représentation graphique d'identifiants

Dans le modèle sémantique élémentaire, nous ne considérons que les identifiants constitués d'un seul attribut.

2.2.6 Sous-type

Supposons qu'il existe deux types d'emprunteurs dans notre bibliothèque universitaire : les membres du personnel et les étudiants. Les membres du personnel et les étudiants sont deux catégories distinctes d'emprunteurs : nous définissons donc deux nouveaux types d'entité `Staff` et `Student` qui sont des **types spécifiques** (ou **sous-types**) de `Subscriber`. `Subscriber` est appelé **type générique** (ou **super-type**).

Les types d'entité spécifiques héritent des propriétés (attributs, identifiants, types d'association) de l'objet générique. Ainsi, dans l'exemple présenté à la figure 2.11, les membres du personnel et les étudiants sont tous deux identifiés par `Number`. Ils participent également au TA `Borrow`.

En plus de ces propriétés communes, ils peuvent également posséder des propriétés qui leur sont propres. Ainsi, dans l'exemple de la figure 2.12, un membre du personnel possède l'attribut : `Laboratory` tandis qu'un étudiant possède l'attribut `Student card`.

Chaque entité spécifique doit obligatoirement appartenir à une et une seule entité générique. Une type générique est qualifié de **disjoint** si, à tout instant, toute entité de celui-ci est en relation avec au plus un de ses sous-types. Il est qualifié de **total** si, à tout instant, toute entité de celui-ci est en relation avec au moins un de ses sous-types. L'ensemble formé par les sous-types est une **partition** si l'ensemble est total et disjoint.

La relation de sous-typage se représente graphiquement par un triangle relié au sur-type par un segment de droite épais et à chaque sous-type par un segment de droite fin (cfr. figure 2.12). Le triangle peut contenir un caractère : **T** pour Total, **D** pour Disjoint et **P** pour Partition.

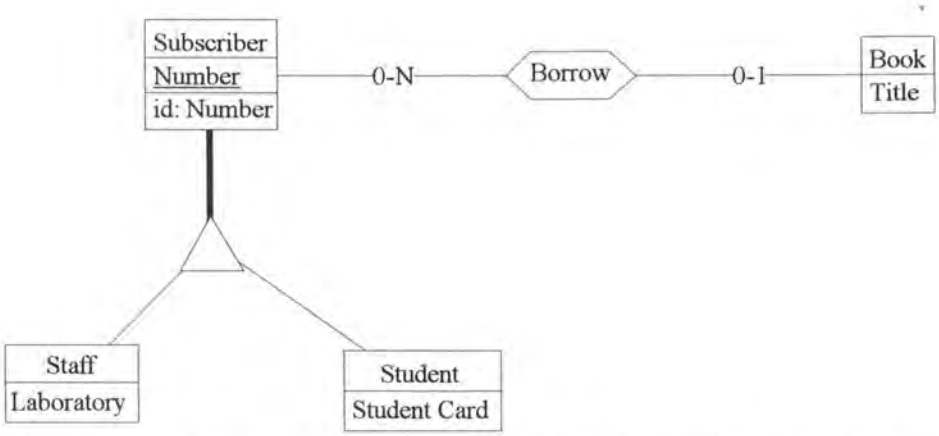


Figure 2.12. Représentation graphique d'une relation de sous-typage

2.3 Modèle sémantique élémentaire

Le **modèle sémantique élémentaire** est une adaptation du modèle NIAM ([VERHEIJEN, 82]) aux formalismes utilisés dans le modèle EA. Le modèle NIAM s'attache à décrire les phrases d'un énoncé en terme d'**objets** jouant un **rôle**. Il repose sur trois concepts : **type d'objet lexical**, **type d'objet non lexical** et **type de lien**.

Dans ce paragraphe, nous commençons par définir les concepts de base du modèle NIAM pour les exprimer dans le formalisme Entité-Association afin de traduire le contenu des textes par des concepts proches du langage naturel. Nous montrons ensuite à partir d'un exemple pourquoi le modèle sémantique élémentaire est bien adapté à la représentation directe des énoncés en langage naturel. Nous montrons enfin comment nous pouvons exprimer les concepts du modèle EA en concepts du modèle sémantique élémentaire.

2.3.1 Concepts de base

2.3.1.1 Types d'objet

Un type d'objet permet de modéliser l'ensemble des objets réels, concrets ou abstraits, ayant des propriétés semblables. Le modèle NIAM distingue les objets non lexicaux désignant des objets autonomes du monde réel (ou **nolot** : Non Lexical Object Type) des objets lexicaux désignant des propriétés (ou **lot** : Lexical Object Type).

Dans le cas de la bibliothèque universitaire, les emprunteurs sont regroupés dans une même classe représentée par le type d'objet non lexical `Subscriber`. De même, l'ensemble des livres est représenté par un type d'objet non lexical `Book`. Par contre, l'ensemble des numéros d'emprunteur est représenté par un type d'objet lexical `Number`.

Un **type d'objet non lexical** est représenté dans le modèle sémantique élémentaire par un TE et dénommé **TE entité** (cfr. figure 2.13).

Figure 2.13. Représentation d'un TE entité dans le modèle sémantique élémentaire

Un **type d'objet lexical** est représenté dans le modèle sémantique élémentaire par un TE contenant un attribut identifiant et dénommé **TE propriété** (cfr. figure 2.14).

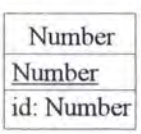


Figure 2.14. Représentation d'un TE propriété dans le modèle sémantique élémentaire

2.3.1.2 Types de lien

Un type de **lien** traduit une classe de lien entre deux types d'objet. Un rôle est associé à un lien. Il a pour but de caractériser sémantiquement le lien. Le modèle NIAM distingue le lien de type **pont de dénomination** et le lien de type **idée**.

Le lien de type **pont de dénomination** établit une relation entre un type d'objet non lexical et un type d'objet lexical. Il exprime le **rôle** « est propriété de ». Dans l'exemple : `A subscriber is characterized by a number`, il existe un pont entre le lot `Subscriber` et le nolot `Number`. Le verbe qui relie le nolot au lot indique le rôle : `is characterized by`. Un lien de type pont est représenté dans le modèle sémantique élémentaire par un type d'association sous forme d'hexagone réduit (cfr. figure 2.15). Il établit une relation entre un TE entité et un TE propriété.

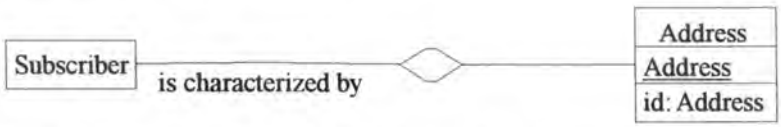


Figure 2.15. Représentation d'un lien de type pont dans le modèle sémantique élémentaire

Le lien de type **idée** établit une relation entre deux types non lexicaux. Il exprime le **rôle** « est associé à ». Dans l'exemple : `A subscriber borrows some books`, il existe un lien de type idée entre `Subscriber` et `Book`, exprimé par le rôle `borrows`. Un lien de type idée est représenté dans le modèle sémantique élémentaire par un type d'association en forme d'hexagone (cfr. figure 2.16). Il établit une relation entre deux TE entités.

Le nom du TA est un nom significatif obtenu à partir des verbes exprimant les rôles associés au TA. Si les deux rôles sont composés du même verbe (ce verbe étant conjugué à la voie active et à la voie passive), il est recommandé de dénommer le TA par le substantif verbal associé à ce verbe.

Figure 2.16. Représentation d'un lien de type idée dans le modèle sémantique élémentaire

2.3.2 Intérêt du modèle sémantique élémentaire

Le modèle sémantique élémentaire est bien adapté à la représentation directe des énoncés exprimés en langage naturel : il associe à chaque verbe de la phrase un rôle et à chaque groupe nominal (groupe sujet, groupe complément) un type d'entité. Il permet donc d'exprimer le contenu des textes sous une forme plus proche du langage naturel que le schéma EA de base.

Considérons, à titre d'illustration, l'énoncé suivant formé de phrases simples.

```
A customer has an address and a name.
A customer works for a department.
A department is occupied by a customer.
A department is characterized by a location.
```

L'expression de ces phrases à l'aide des concepts du modèle sémantique élémentaire conduit au schéma suivant :

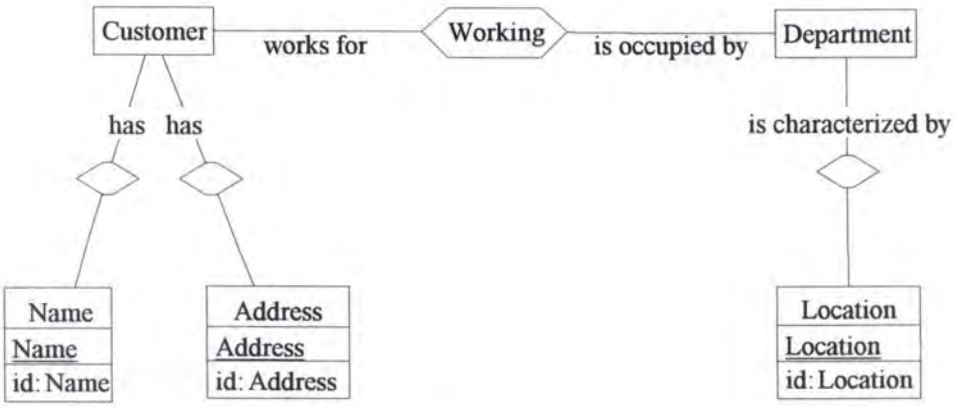


Figure 2.17. Modèle sémantique élémentaire

2.3.3 Expression des concepts

Dans ce paragraphe, nous expliquons comment nous avons exprimé les concepts du modèle EA de base en concepts du modèle sémantique élémentaire. Cette expression se traduit en un enrichissement du modèle sémantique élémentaire par la prise en compte du concept de **cardinalité**.

Pratiquement, nous montrons comment nous avons intégré le concept de cardinalité dans le modèle sémantique élémentaire et comment nous avons transformé le modèle EA de base afin de mettre en évidence les concepts **TE entité, TE propriété, pont de dénomination** et **idée**. Ces transformations portent sur les attributs et sont extraites de [HAINAUT, 94] et [DBMAIN, 95].

Toutes les transformations sont à **sémantique constante** ce qui signifie que les transformations modifient la syntaxe d'un schéma mais pas sa sémantique : le schéma EA de base et le schéma sémantique élémentaire décrivent le même domaine d'application.

2.3.3.1 Expression d'un TE

Un type d'entité du modèle EA correspond à un **TE entité** dans le modèle sémantique élémentaire.

2.3.3.2 Expression d'un attribut

Un attribut (multivalué ou monovalué ; facultatif ou obligatoire) est exprimé par un **TE propriété**. Pour obtenir un schéma équivalent, nous procédons à la transformation des attributs en type d'entité. Pour transformer un attribut `A` caractérisé par un couple de valeurs `[i-j]`, nous procédons par représentation des valeurs : nous créons un nouveau type d'entité `TA` (exprimant un **TE propriété**) possédant un attribut `A` monovalué. Nous relions ensuite `TA` à `E` par un type d'association **(pont de dénomination)**. Les cardinalités associées au rôle joué par `TA` dans l'association sont `1-N` et celles associées au rôle joué par `E` sont `i-j`.

Dans la phrase : « `A subscriber can have at most 5 address` », le modèle Entité-Association interprète `subscriber` comme un type d'entité et `address` comme attribut de ce type d'entité. Par contre, le modèle sémantique élémentaire établit une relation de type **pont de dénomination** (exprimée par le verbe `has`) entre un **TE entité** (`subscriber`) et **TE propriété** (`address`) (cfr. figure 2.18).

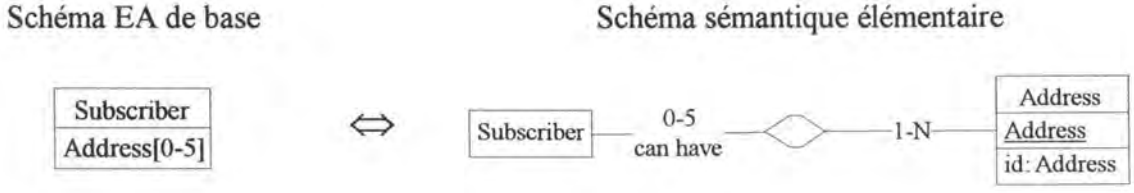


Figure 2.18. Expression d'un attribut simple

2.3.3.3 Expression d'un attribut identifiant

Un attribut identifiant est exprimé également par un **TE propriété.** Pour transformer un attribut identifiant, nous procédons également par représentation des valeurs.

Considérons, par exemple, la phrase : « `A subscriber is identified by a name` ». Le modèle Entité-Association interprète `name` comme un attribut identifiant du TE `subscriber`. Par contre, le modèle sémantique élémentaire interprète `subscriber` comme un TE exprimant un **TE entité** et `name` comme un **TE propriété**. La cardinalité du rôle joué par le **TE propriété** est `1-1` (cfr. figure 2.19).

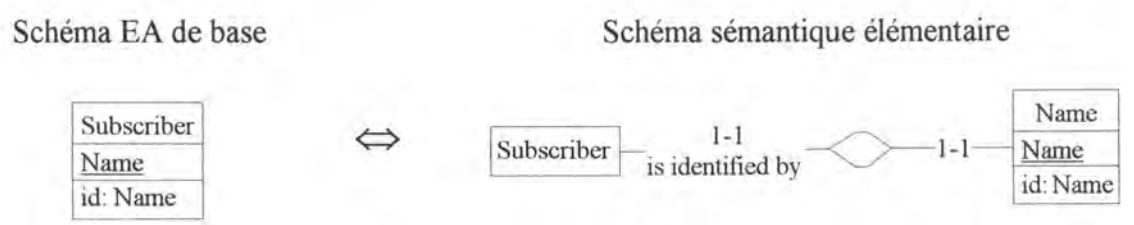


Figure 2.19. Expression d'un attribut identifiant

2.3.3.4 Expression d'un type d'association

Un **lien de type idée** correspond au type d'association binaire du modèle Entité-Association et repose sur la définition du type d'association définie au paragraphe 2.2.4. Les contraintes d'intégrité associées à un lien de type idée sont donc les cardinalités associées aux rôles du TA.

2.3.3.5 Expression d'un sous-type

Nous considérons également qu'un **TE entité** peut être déclaré comme **sous-type** d'un autre. La relation qui les lie est alors la relation de sous-typage telle qu'elle a été définie au paragraphe 2.2.6.

3. Démarche

3.1 Introduction

Au premier chapitre, nous avons caractérisé notre **méthode** de conception d'un schéma conceptuel par trois composants de base : une **démarche** fondée sur des **modèles** et exploitée par des **outils logiciels** (cfr. figure 3.1).

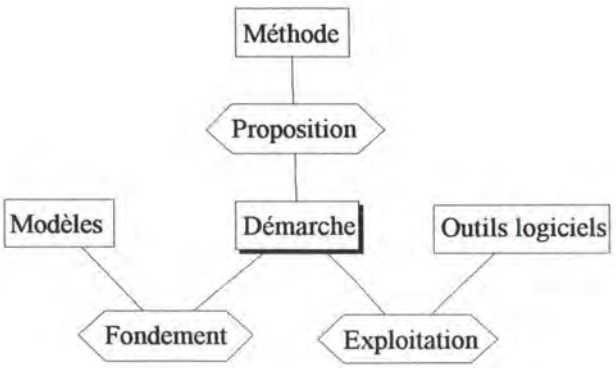


Figure 3.1. Méthode et démarche

Dans ce chapitre, nous proposons une démarche formée d'**étapes** et de **règles** en vue de maîtriser les étapes de la conception d'un schéma conceptuel à partir d'un énoncé en langage naturel. L'enchaînement des différentes étapes élémentaires du processus d'élaboration du schéma conceptuel est présenté dans la figure 3.2.

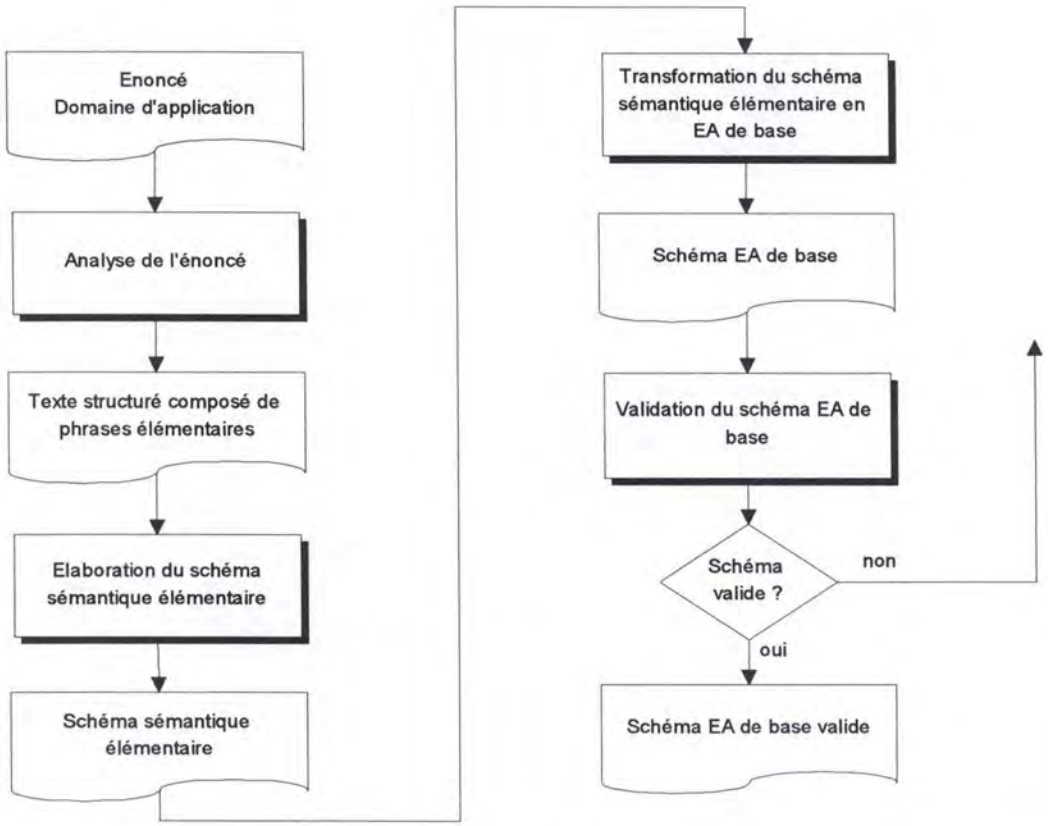


Figure 3.2. Etapes de notre démarche

3.2 Analyse de l'énoncé

Cette étape est consacrée à l'organisation du contenu informationnel de l'énoncé sous la forme d'un texte structuré (cfr. figure 3.3).

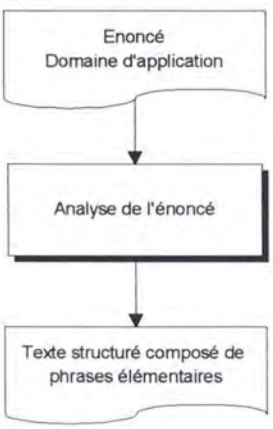


Figure 3.3. Analyse de l'énoncé

L'analyste dispose rarement d'un énoncé structuré pour élaborer un schéma conceptuel. Il le construit très souvent à partir de plusieurs documents (rapports d'entretiens, textes de procédure administrative ou de gestion, etc.). Généralement ces documents sont partiellement redondants et peuvent présenter des incohérences. La construction directe d'un schéma à partir

de tels documents risque d'aboutir à un schéma conceptuel médiocre. Pour éviter d'aboutir à un tel résultat, nous proposons de décomposer l'énoncé sous la forme d'un **texte structuré** de **phrases élémentaires**.

3.2.1 Phrase élémentaire

Une phrase élémentaire est un système de propositions et de mots qui ne peut être décomposé en constructions plus courtes sans perte de sens : elle est sémantiquement irréductible ([BODART, 96]). La structure d'une phrase élémentaire est la suivante :

<sujet> <verbe> <complément>

Exemple

La phrase (1) est une phrase élémentaire.

(1) `Ann employs Bob.`

(2) `Ann employs Bob who works in Brussels.`

La phrase (2) n'est pas élémentaire. Elle peut être décomposée en deux phrases élémentaires : `Ann employs Bob; Bob works in Brussels.`

3.2.2 Forme abstraite d'une phrase élémentaire

Une phrase élémentaire sera exprimée sous sa forme abstraite. Elle se rapportera à des classes de faits de façon à exprimer leur formalisation sous la forme de type d'objet ou de type de lien.

Exemple

La phrase (2) sera décrite comme suit :

(3) `A person employs a person.`

Une phrase élémentaire ne peut contenir des conjonctions logiques.

Exemple

La phrase (4) n'est pas élémentaire.

(4) `A subscriber has a name and an address.`

La phrase (4) contient une conjonction logique. Elle peut être décomposée en deux phrases élémentaires : `A subscriber has a name; A subscriber has an address.`

Cependant, **afin d'éviter un style trop télégraphique,** la conjonction logique **and** peut être utilisée lorsqu'elle permet d'alléger le texte. Nous acceptons donc une phrase dont la structure est la suivante :

<sujet> <verbe> <complément> [(**and**) <complément>]

Par extension, nous considérons une phrase de cette structure comme phrase élémentaire.

Exemple

La phrase (5) n'est pas élémentaire mais néanmoins acceptée.

(5) `An author has a name and an address.`

D'autre part, chaque phrase élémentaire doit posséder une **sémantique explicite** ; elle ne doit pas représenter des raccourcis ou des structures implicites. A cette fin, l'emploi des pronoms est proscrit en faveur des types d'objets qu'ils représentent.

Exemple

Les phrases (6) et (7) sont élémentaires.

(6) `A book has a ISBN-number.`

(7) `It's written by an author.`

La phrase (7) contient un pronom qui fait référence à la phrase (6). Afin de supprimer toute ambiguïté, la phrase (7) doit être remplacée par `A book is written by an author.`

3.2.3 Expression de contrainte d'intégrité

Une phrase élémentaire peut avoir pour rôle de limiter la signification d'autres phrases élémentaires. Dans ce cas, elle est source de contrainte d'intégrité.

Exemple

La phrase élémentaire suivante précise le sens apporté à la phrase (3) :

(8) `A person employs at most five persons.`

La phrase (8) sera la source d'une contrainte de cardinalité maximum du rôle joué par `person`.

3.3 Elaboration du schéma sémantique élémentaire

La production du schéma sémantique élémentaire se fait par transformation du texte structuré issu de l'étape précédente (cfr. figure 3.4).

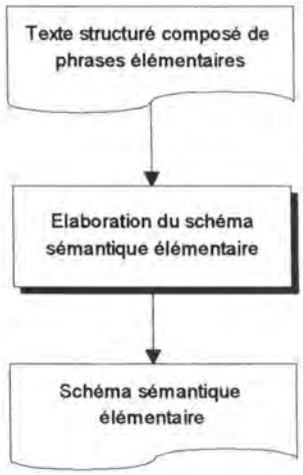


Figure 3.4. Elaboration du schéma sémantique élémentaire

Cette phase consiste à transformer les éléments d'une phrase élémentaire en objets formels du modèle conceptuel. Il s'agit de déterminer quels sont les mots à représenter par des TE entités ou des TE propriétés et de déterminer les relations qui les lient.

Considérons l'énoncé suivant formé de phrases simples mises sous forme abstraite.

```
(1)  A subscriber has a name.
(2)  A subscriber borrow books.
(3)  A book is characterized by an a title.
```

Pour construire le schéma sémantique élémentaire, nous procédons de façon progressive afin de mettre en évidence pour chaque phrase additionnelle prise en compte les éléments nouveaux qu'elle peut contenir : TE entité, TE propriété (exprimé par un TE contenant un attribut identifiant), type de lien et contraintes d'intégrité relatives à des éléments préalablement définis.

La phrase (1) fait apparaître un TE entité `subscriber`, un TE propriété `name` et un lien de type pont de dénomination qui les associe. Dans la phrase (2), nous reconnaissons un nouveau TE entité `books` et un lien de type idée qui associe `books` à `subscriber`. Dans la phrase (3), nous reconnaissons un TE propriété `title` associé à `books` par un lien de type pont de dénomination.

Cet exemple peut être représenté par le schéma de la figure 3.5. Le contenu du texte structuré n'est pas suffisamment précis pour estimer les cardinalités associées aux rôles. Nous avons donc attribué aux cardinalités des valeurs par défaut (cfr. chapitre 2, paragraphe 2.2.4.3).

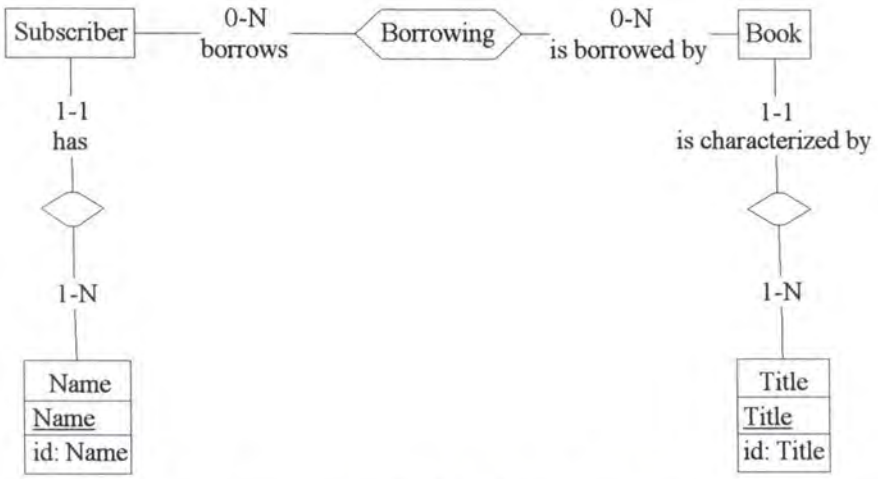


Figure 3.5. Résultat de la transformation d'un texte structuré en schéma sémantique élémentaire

Le rôle du langage dans l'activité d'abstraction a orienté notre étude vers un formalisme des mécanismes linguistiques (cfr. chapitres 4 et 5) utilisés par l'analyste pour traduire les éléments d'une phrase en concepts. Nos outils logiciels (cfr. chapitre 6) incluent ces mécanismes et permettent ainsi une automatisation de cette phase.

3.4 Transformation du schéma sémantique élémentaire en schéma Entité-Association

L'objectif de cette phase est de transformer le schéma sémantique élémentaire en schéma Entité-Association de base (cfr. figure 3.6). Cette transformation est réalisée à partir de règles de transformation de schéma établies au chapitre 2. Pour passer d'un schéma sémantique élémentaire à un schéma EA de base, il suffit de transformer les TE propriétés et les TE entités exprimant des concepts représentables sous la forme de TA simples (c'est-à-dire des TA binaires et sans attribut). Notons que ces transformations sont à **sémantique constante** ce qui signifie que les transformations modifient la syntaxe d'un schéma mais pas sa sémantique (cfr. chapitre 2, paragraphe 2.3).

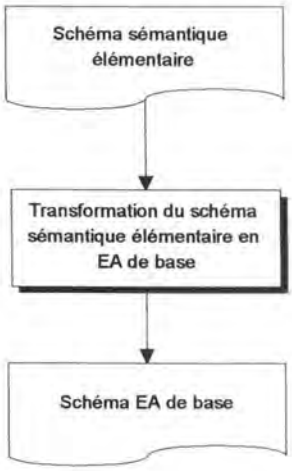


Figure 3.6. Transformation du schéma sémantique élémentaire en EA de base

3.4.1 Transformation d'un TE propriété en attribut

Soit B un TE propriété. Il est donc constitué d'un attribut identifiant B1, associé à un type d'entité A via un type d'association ; les cardinalités du rôle joué par B dans le cadre du type d'association sont [1-N] ou [1-1]. Cette situation pourrait se présenter graphiquement comme suit :

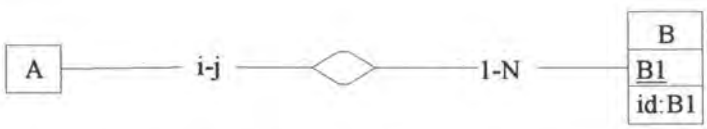


Figure 3.7. Schéma sémantique élémentaire

L'objectif de la transformation est d'éliminer le TE B en le remplaçant par un attribut. Pour transformer le TE B en attribut, nous procédons comme suit :

1. nous représentons le TE B par un attribut B1' de même type que B1 que nous ajoutons à A ;
2. nous caractérisons l'attribut B1' par le couple de valeurs [i-j] ;
3. si les cardinalités du rôle joué par B sont [1-1], alors B1' doit être déclaré identifiant de A.

Le schéma EA de base résultant de cette transformation est celui de la figure 3.8.

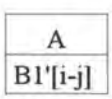


Figure 3.8. Transformation d'un TE en attribut

3.4.2 Transformation d'un TE entité en TA simple

Le schéma sémantique élémentaire peut contenir des TE entités exprimant un concept représentable sous la forme d'un TA simple. Afin d'obtenir une représentation conceptuelle plus **significative** du réel perçu, nous transformons les TE entités qui ont la signification d'un TA.

Soit `TE`, un TE entité exprimant un concept représentable sous la forme d'un TA simple. Le TE entité `TE` est alors associé à deux autres TE entités `A`, `B` et à aucun TE propriété. Les cardinalités des rôles joués par le TE `TE` sont [1-1]. Les cardinalités jouées par `A` ou `B` dans le cadre des TA qui les unissent à `TE` sont [0-1] ou [1-1]. Cette situation pourrait se présenter graphiquement comme suit :

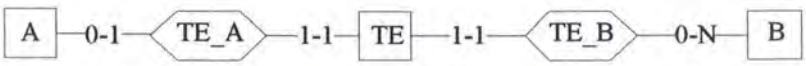


Figure 3.9. TE entité associé à deux TE entités

L'objectif de la transformation est d'éliminer le TE `TA` en le remplaçant par un TA de même nom. Le schéma EA de base résultant de cette transformation est celui de la figure 3.10.

Figure 3.10. Transformation d'un TE entité en TA

Pour faciliter la compréhension de ce mécanisme, considérons l'exemple illustré par la figure 3.11.

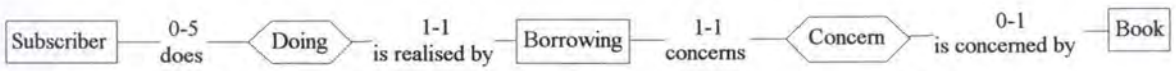


Figure 3.11. Exemple de TE entité représentant un TA

Dans cet exemple, le TE entité `Borrowing` exprime un concept représentable sous la forme d'un TA. La transformation du TE entité `Borrowing` en TA aboutit au schéma équivalent de la figure 3.12.

Figure 3.12. Exemple de transformation d'un TE entité en TA

3.5 Phase de validation

La phase de validation (cfr. figure 3.13) utilise des **règles formelles** destinées à vérifier si les spécifications obtenues lors des phases précédentes sont correctes. Ces règles doivent permettre de vérifier que chaque classe d'objets repris dans le schéma EA de base possède bien, en fonction de son type, toutes les propriétés prévues dans le modèle EA. Par exemple, un TE devrait comporter les éléments suivants : au moins un attribut et au moins un identifiant.

L'application de ces règles ne fournit qu'une **validation formelle**. Il est dès lors utile de compléter celle-ci par la **validation du contenu**. Il s'agit, notamment, de s'assurer que le schéma est bien la représentation fidèle du domaine d'application.

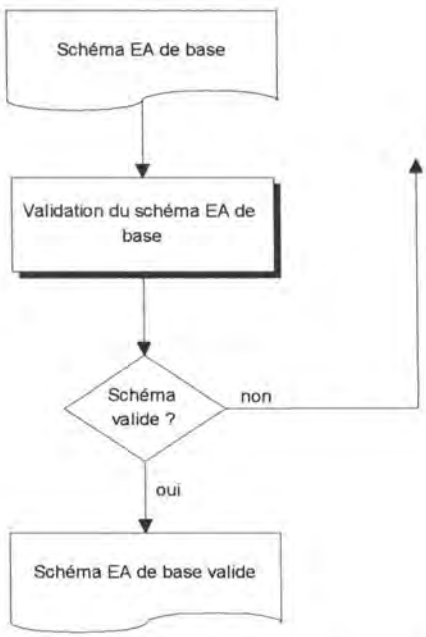


Figure 3.13. Validation du schéma EA de base

3.5.1 Validation formelle

La validation formelle est obtenue par application de tests formels. Ces tests sont appliqués aux concepts de TE, TA et de cardinalités. Il importe de tenir compte du fait qu'il s'agit de tests indicatifs et non généraux. Ils détectent les **structures à problème** suivantes : TE sans attribut ou sans identifiant, TA similaires et cardinalités indéterminées pendant la phase d'élaboration du schéma sémantique élémentaire.

3.5.1.1 TE sans attribut ou sans identifiant

Un **TE ne possédant pas d'attribut** peut être considéré comme présomption de spécification incomplète. En effet, une telle absence signifie que l'existence possible d'une entité est la seule

information que nous voulons exploiter. De telles situations sont extrêmement rares ([BODART, 96]). Cette règle générale n'est cependant pas applicable au cas des sous-types. Pour ceux-ci, l'absence totale d'attribut n'est pas un élément d'incomplétude puisqu'ils héritent des propriétés de leur sur-type.

D'autre part, un **TE sans identifiant** peut être également considéré comme une structure à problème car l'identifiant constitue un mécanisme de représentation privilégié de représentation d'un TE : il permet de repérer univoquement chaque occurrence du type.

3.5.1.2 TA similaires

Des **TA similaires** sont des TA sémantiquement équivalents mais représentés sous la forme de TA distincts. Ils sont définis sur les mêmes TE et expriment une relation sémantiquement équivalente. Pour détecter la présence de TA similaires, il faut vérifier les TA qui relient au moins deux TE identiques.

Considérons l'exemple représenté par la figure 3.14.

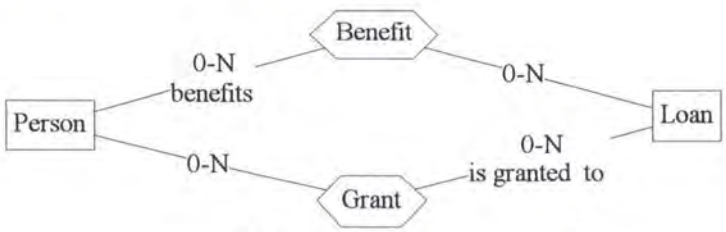


Figure 3.14. Exemple de TA similaires

Les TA `Benefit` et `Grant` représentent la même relation entre les TE `Person` et `Loan` : ils définissent le prêt accordé à une personne. La relation doit donc être représentée par un seul TA dont le rôle joué par `Person` est `benefits` et celui joué par `Loan` est `is granted to`.

3.5.1.3 Cardinalités indéterminées pendant la phase d'interprétation

Il est extrêmement rare que le contenu du texte structuré soit suffisamment précis pour permettre d'estimer directement toutes les cardinalités. Par exemple, La phrase élémentaire « `A subscriber borrows books` » ne contient pas assez d'informations pour estimer les cardinalités associées aux rôles.

Il est dès lors nécessaire d'enrichir la connaissance du domaine d'application. Ainsi, les phrases élémentaires « `A subscriber can borrow at most five books` » et « `A book can be borrowed by only one subscriber` » constituent une source d'information suffisante pour fixer les cardinalités des rôles assumés par `subscriber` et `book` (cfr. figure 3.15).

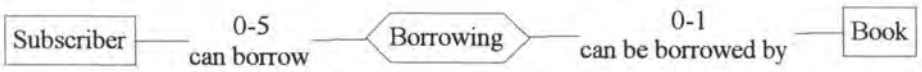


Figure 3.15. Schéma EA et cardinalités déterminées

3.5.2 Validation du contenu

La validation du contenu consiste à s'assurer que le schéma est bien la représentation fidèle du domaine d'application. Il importe en effet de contrôler l'aptitude du schéma EA de base à réaliser les objectifs informationnels que poursuit l'organisation.

Pour contrôler cette aptitude, l'organisation accordera une attention particulière au contenu sémantique du schéma EA de base. Elle vérifiera notamment que toutes les informations qu'elle manipule sont bien reprises dans le schéma.

Pour aider les membres de l'organisation qui ne maîtrisent pas le modèle EA de base, nous proposons d'utiliser NATURAL, le **paraphraseur de schémas** développé par [DEFLORENNE, 96] (cfr. chapitre 1, paragraphe 1.3.3.1).

Pour favoriser la participation active des membres de l'organisation à l'évaluation des spécifications, nous proposons d'utiliser également des **outils de prototypage**. Ces outils permettent une vérification expérimentale du contenu sémantique du schéma, sans devoir attendre son implantation finale. Ils reposent sur l'idée que la façon la plus immédiate pour une organisation de prendre connaissance des spécifications du schéma conceptuel nouveau est l'expérimentation directe de la solution proposée.

4. Approche linguistique

4.1 Introduction

Dans ce chapitre, nous présentons l'**approche linguistique** sur laquelle est basée notre démarche (cfr. figure 4.1). Cette approche est fondée sur une tentative de formalisation des mécanismes intellectuels par lesquels l'analyste est capable d'abstraire la réalité perçue en termes de concepts.

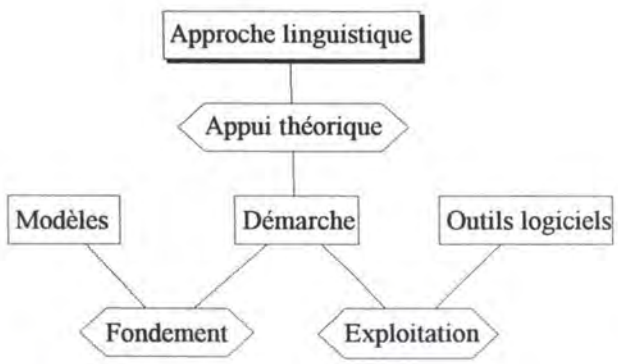


Figure 4.1. Méthode et démarche basée sur une approche linguistique

Dans le paragraphe 4.2, nous tentons de comprendre les **mécanismes linguistiques** utilisés par un analyste qui, à partir d'un texte, construit un schéma conceptuel. Le troisième paragraphe expose les fondements de l'approche linguistique basée sur **la théorie des cas de Fillmore**. Le quatrième paragraphe montre de quelle manière nous avons adapté cette approche.

4.2 Mécanismes linguistiques

Afin de mettre en évidence les mécanismes utilisés par un analyste, [PROIX, 89] étudie sur quelques exemples ce qui lui semble être sa démarche. Nous supposons que le modèle conceptuel utilisé pour modéliser le domaine d'application est le modèle Entité-Association. Il s'agit donc pour l'analyste d'identifier les phénomènes réels à représenter à l'aide de concepts de type d'entité, d'attribut et d'identifier les différentes associations entre ces types d'entité.

4.2.1 Description des mécanismes

Supposons que l'analyste rencontre la phrase suivante :

```
A subscriber has a name and an address.
```

Il détermine immédiatement qu'il existe un type d'entité `subscriber` dont les attributs sont `name` et `address`. Il aboutit donc au schéma EA de la figure 4.2.

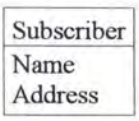


Figure 4.2. Schéma EA représentant le concept d'emprunteur

Pour aboutir à un tel résultat, Proix propose la réponse suivante : *l'analyste connaissait déjà la solution*. Il avait déjà rencontré ces concepts antérieurement. Il savait que le mot `subscriber` représente un type d'entité et que les mots `name` et `address` représentent des attributs. Il a travaillé par référence à son propre savoir et mis en œuvre une **connaissance de type lexical**.

Supposons maintenant que l'analyste rencontre la phrase suivante (exemple tiré de [ROLLAND, 91]) :

```
A flower has a pedistylus and a folicul.
```

Dans ce cas-ci, il ne connaît rien à la sémantique des mots (il n'est pas botaniste). Il peut cependant émettre l'hypothèse que le mot `flower` représente un objet qui est composé des objets représentés par les mots `pedistylus` et `folicul`. Il peut donc proposer le schéma de la figure 4.3.

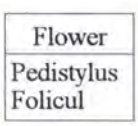


Figure 4.3. Schéma EA représentant le concept de fleur

Cette fois, le propre savoir de l'analyste n'a pu être utilisé puisqu'il ne connaît pas la sémantique des mots. La non connaissance de la signification des mots n'a cependant pas empêché l'analyste d'émettre une hypothèse qui paraît correcte. En fait, son raisonnement semble être basé sur la connaissance du langage, et plus précisément sur la reconnaissance d'un schéma de phrase familier :

<sujet> <verbe exprimant l'état> <complément>

Cette connaissance du langage lui permet d'associer le groupe sujet à un type d'entité et les mots du groupe complément à des attributs. Pour aboutir à un tel résultat, l'analyste a fait appel à :

1. une connaissance grammaticale ;
2. une connaissance lexicale du verbe utilisé dans la phrase.

4.2.2 Intérêt des mécanismes

L'activité intellectuelle mise en œuvre par l'analyste au cours de la phase d'élaboration d'un schéma conceptuel repose sur deux mécanismes :

1. une reconnaissance de concepts acquis antérieurement ;
2. une reconnaissance de schémas particuliers du langage ; schémas que l'analyste a appris antérieurement à associer à des situations du réel.

Le premier de ces mécanismes met en œuvre une importante connaissance de type lexical. Dans la perspective d'un outil d'aide à la conception, la formalisation de ce mécanisme paraît difficile : il faudrait en effet constituer d'énormes dictionnaires par domaine d'application ([PROIX, 89]).

Le second mécanisme fait appel à un ensemble de connaissances grammaticales sur le langage, à un ensemble de connaissances lexicales des verbes et un ensemble de correspondances entre structures de phrase et situations réelles. Ce mécanisme se prête mieux à une telle formalisation, surtout si nous considérons que le langage utilisé par l'analyste est un sous-ensemble élémentaire de la langue anglaise. L'ensemble des structures à formaliser est donc relativement restreint. Quant aux verbes, leur nombre reste relativement limité pour juger raisonnable d'en constituer un dictionnaire ([PROIX, 89]).

Pour ces raisons, nous basons notre démarche linguistique sur une typologie des phrases permettant de connaître leur sémantique. Dans le paragraphe suivant, nous présentons les fondements de l'approche linguistique basée sur la théorie des cas de Fillmore et nous montrons comment nous avons adapté cette approche à la conception des schémas conceptuels.

4.3 Fondements de l'approche linguistique

Notre approche linguistique est une adaptation de la **théorie des cas** développée par Fillmore. De nombreux chercheurs se sont inspirés de cette théorie pour interpréter les textes décrivant un schéma conceptuel. Parmi ceux-ci, citons [PROIX, 89], [KERSTEN, 87] et [BUCHHOLZ, 96].

4.3.1 Théorie des cas de Fillmore

Fillmore[2] a mis en évidence l'importance des **verbes** et des **rôles** (**cases**) tenus par les différents constituants des phrases dans la compréhension de ces phrases. Il soutient qu'il est possible

[2] Nous renvoyons le lecteur à [SABAH, 90a] pour une développement plus complet de la théorie des cas de Fillmore.

d'identifier un ensemble de rôles sémantiques permettant de mettre en évidence les relations sémantiques qui existent entre les noms et le verbe dans une phrase simple.

Ses argumentations se fondent sur les principes suivants :

1. les groupes de mots d'une phrase simple entretiennent une relation particulière, appelée **rôle,** avec le verbe ;
2. la sémantique d'un texte est exprimée par la sémantique du verbe qui le compose, mais aussi par la sémantique associée aux rôles qui apparaissent dans le texte.

4.3.1.1 Rôles sémantiques proposés par Fillmore

Nous pouvons résumer les principaux rôles mis en évidence par Fillmore :

1. **rôle agent** : ce rôle est associé aux groupes de mots représentant le ou les individus réalisant l'action énoncée par le verbe ;
2. **rôle instrument** : ce rôle est associé aux groupes de mots représentant le ou les objets à l'aide duquel l'action est réalisée ;
3. **rôle datif** : ce rôle est associé aux groupes de mots représentant le ou les objets affectés par l'action ou l'état décrit par le verbe de la préposition ;
4. **rôle lieu** : ce rôle est associé aux groupes de mots exprimant la localisation spatio-temporelle de l'action ou de l'état décrit par le verbe de proposition ;
5. **rôle factitif** : ce rôle est associé aux groupes de mots représentant des objets produits par l'action.

La phrase suivante permet d'illustrer ces rôles (exemple tiré de [SABAH, 90a]) :

```
John opens the door with his key.
```

Le mot `John` est associé au rôle agent, le mot `door` au rôle objet et le mot `key` au rôle instrument.

Il résulte des études de Fillmore que nous pouvons associer à n'importe quel mot du langage un ensemble de rôles qu'il est capable de tenir. Cet ensemble est défini par rapport au sens du mot.

Par exemple, le mot `key` peut être associé aux rôles : objet, datif, factitif, instrument et agent :

(1) `John takes the key.`	(objet)
(2) `The key is big.`	(datif)
(3) `The locksmith makes a key.`	(factitif)
(4) `John opens the door with his key.`	(instrument)
(5) `The key opens the door.`	(agent)

Dans chacune des phrases précédentes, le rôle associé au mot `key` est déterminé en fonction d'une part du sens du verbe et d'autre part du statut grammatical du mot dans la phrase. Ainsi, bien que le mot `key` soit le sujet des phrases (2) et (5), il n'est pas associé au même rôle. Les

rôles sont en effet déterminés par rapport au sens du verbe : un verbe d'état pour la phrase (2) et un verbe d'action pour la phrase (5).

4.3.1.2 Structure profonde d'une phrase

Le **verbe** d'une phrase est donc l'élément autour duquel s'organise la structure grammaticale mais c'est aussi **l'élément déterminant de la sémantique de la phrase.** Pour un verbe donné, dans une phrase donnée, un seul groupe nominal peut être lié à ce verbe par un rôle sémantique donné (la théorie de Fillmore ne tient pas compte des phénomènes complexes tels que la coordination et cela vaut bien entendu uniquement pour les phrases « simples »).

Pour Fillmore, il est possible d'attacher **à priori** à chaque verbe ses rôles sémantiques possibles et donc de construire une structure mettant en évidence le type de lien qui lie le verbe et les différents mots de la phrase. Ainsi, la phrase (2) pourrait être caractérisée par la structure suivante (S1):

<groupe sujet (rôle datif)> <groupe verbal exprimant un état>

Cette structure, généralement appelée par les linguistes **structure profonde**, permet de comprendre la phrase. C'est en effet par reconnaissance de la structure profonde d'une phrase que nous sommes capables de la comprendre.

Les études de Fillmore ont montré qu'il est possible de définir un certain nombre de standards parmi ces structures profondes de phrases. Par exemple, la description d'un état se fait par l'intermédiaire de phrases respectant la structure S1 énoncée précédemment. La phrase « `The man is tall.` » respecte également cette structure.

La construction de ces standards montre que tous les rôles sémantiques ne peuvent être associés à n'importe quel verbe, et que nous pouvons classifier les structures standards en fonction de la sémantique du verbe qu'elles contiennent.

La compréhension d'une phrase passe donc par une reconnaissance des structures standards. C'est donc sur base de la reconnaissance du sens du verbe que nous sommes capables d'associer les rôles qui lui conviennent, de reconnaître une structure standard et finalement de comprendre la sémantique de la phrase ([PROIX, 89]).

4.3.1.3 Limites de la théorie de cas

La théorie des cas est bien adaptée à la compréhension de **phrases simples** (sujet, verbe, complément d'objet, complément de phrase) mais n'est pas utilisable telle qu'elle pour la compréhension de **phrases complexes** (des phrases contenant des propositions subordonnées, par exemple). Elle laisse en effet un certain nombre de questions en suspens : quelles sont les relations sémantiques entre les propositions de la phrase ? Les caractéristiques de la phrase influent-elles sur la détermination du rôle d'une proposition ?

Des chercheurs ont adapté la théorie des cas à la compréhension des phrases complexes en étendant la notion de rôles aux propositions. Cette adaptation les a amenés à identifier le rôle joué par chaque proposition subordonnée par rapport à la proposition principale [SABAH, 90a].

Enfin, d'autres questions sont liées à la difficulté de justifier un ensemble cohérent de rôles sémantiques. Le **choix a priori** d'un certain nombre de rôles sémantiques n'est en effet pas neutre vis-à-vis des interprétations ultérieures de la structure construite ([SABAH, 90a]).

Ainsi, par exemple, la structure profonde attachée à un verbe entraîne que seuls certains aspects de la situation décrite vont être pris en compte.

Dans la phrase « `John cuts down the three with an axe.` ». Si `cuts` est interprété (à priori) comme un verbe d'action correspondant à la structure suivante :

<groupe sujet (rôle agent)> <groupe verbal exprimant une action> <groupe complément (rôle objet) >

il est alors impossible de déterminer le rôle sémantique de `axe`.

4.3.2 Particularités de notre approche

Dans le contexte de conception de schémas conceptuels, l'analyse sémantique des textes peut être simplifiée par rapport aux situations décrites par Fillmore. La représentation conceptuelle que l'analyste fait du domaine d'application est une image pauvre de celui-ci. Il s'agit donc d'atteindre une compréhension suffisante du domaine d'application pour identifier les concepts du modèle conceptuel. Cette relative pauvreté de la compréhension recherchée des textes nous a permis de mettre en évidence un ensemble raisonnable de rôles sémantiques.

En accord avec notre méthode, les textes analysés sont des textes composés de **propositions élémentaires**. La grammaire de ces propositions peut se résumer comme suit : elles sont affirmatives et déclaratives ; elles sont composées d'un verbe et d'un ensemble de groupes de mots, chacun remplissant un certain rôle (le **case** de Fillmore) par rapport à ce verbe.

Tenant compte de ces réductions de la langue naturelle, nous avons adapté la théorie des cas de Fillmore à la conception de schémas conceptuels. Cette adaptation porte essentiellement sur la définition d'un ensemble de rôles spécifiques et par conséquent d'un ensemble de schémas standards adaptés aux phrases élémentaires.

4.4 Typologies des rôles et des verbes

4.4.1 Typologie des rôles

Partant des rôles définis par Fillmore et Proix, nous avons retenu un ensemble de rôles spécifiques à la construction d'un schéma sémantique élémentaire. L'ensemble des rôles retenus est le suivant : {POSSESSEUR, PROPRIETE, IDENTIFIANT, GENERALISATION, SPECIALISATION, ACTEUR, OBJET, CONTRAINT, CONTRAINTE}.

POSSESSEUR est attribué aux mots apparaissant comme propriétaire d'objets dans les phrases. **PROPRIETE** est attribué aux mots apparaissant comme des objets possédés par d'autres objets.

Exemple

`A book has a title.`

POSSESSEUR est attribué au mot `book` et le rôle PROPRIETE au mot `title`.

IDENTIFIANT est attribué aux mots apparaissant comme des objets identifiants d'autres objets. La présence du rôle PROPRIETE ou du rôle IDENTIFIANT dans une phrase implique la présence du rôle POSSESSEUR dans la même phrase et réciproquement.

Exemple

`A book is identified by its ISBN number.`

POSSESSEUR est attribué au mot `book` et le rôle IDENTIFIANT au mot `ISBN number`.

SPECIALISATION est attribué aux mots apparaissant comme une spécialisation d'un objet générique de plus haut niveau. **GENERALISATION** est attribué aux mots apparaissant comme une classe d'objets génériques. La présence du rôle SPECIALISATION dans une phrase implique la présence du rôle GENERALISATION dans la même phrase et réciproquement.

Exemple

`A road can be a motorway or a national road.`

SPECIALISATION est attribué aux mots `motorway` et `national road` et le rôle GENERALISATION au mot `road`.

ACTEUR est attribué aux mots décrivant les phénomènes exécutant les actions décrites dans les phrases. **OBJET** est attribué aux mots décrivant les phénomènes subissant les actions décrites dans les phrases.

Exemple

`A subscriber borrows a book.`

ACTEUR est attribué au mot `subscriber` et le rôle OBJET au mot `book`.

CONTRAINTE est attribué aux mots décrivant les phénomènes qui contraignent d'autres phénomènes du monde du réel. Ces mots sont généralement des « mots-clés » (par exemple, `only`, `at least`, `at most`, `maximum`, etc.). **CONTRAINT** est attribué aux mots décrivant les phénomènes qui sont contraints par d'autres phénomènes.

Exemples

`A subscriber borrows only one book.`
CONTRAINTE est attribué au groupe de mots `only one`. Les rôles CONTRAINT et ACTEUR sont attribués au mot `subscriber` ; le rôle OBJET au mot `book`.

`A subscriber can borrow a book.`
CONTRAINT et ACTEUR est attribué au groupe de mots `subscriber` (la source de contrainte est l'auxiliaire `can`) ; le rôle OBJET au mot `book`.

4.4.2 Typologie des verbes

Partant des rôles définis précédemment, nous avons retenu quatre classes de verbe :

1. **Les verbes de description** sont les verbes employés pour décrire la structure des objets du monde réel. Ces verbes expriment une notion simple de possession. Par exemple, `have`, `to be characterized` font partie de cette classe.
2. **Les verbes d'identification** sont les verbes utilisés pour indiquer les identifiants des objets du monde réel. Par exemple, `to be identified` fait partie de cette classe.
3. **Les verbes de spécialisation** sont les verbes utilisés pour exprimer une relation de spécialisation entre des objets du monde du réel. Par exemple, le verbe `to be` fait partie de cette classe.
4. **Les verbes d'action** sont des verbes utilisés pour décrire une relation entre objets du monde réel. Ces verbes sont étroitement liés au domaine d'application.

4.5 Schémas standards

A partir de l'analyse des rôles et des classes de verbe, nous avons pu définir un ensemble de schémas standards. Ces schémas standards représentent des structures de **phrases simples**. Nous avons défini deux grandes catégories de schémas élémentaires :

1. **les schémas structuraux** qui concernent la description des entités ;
2. **les schémas associatifs** qui concernent les relations entre ces entités.

4.5.1 Schémas structuraux : SS

Ces schémas définissent toutes les structures de rôles associées à des situations de description d'objets du monde réel. Nous distinguons deux sous catégories selon qu'il s'agit de schémas comportant un verbe de la catégorie DESCRIPTION (SSS) ou qu'il s'agit de schémas comportant un verbe de la catégorie IDENTIFIANT (SSI).

4.5.1.1 Schémas structuraux simples (SSS)

Ils comportent les éléments suivants :

1. un verbe de la catégorie DESCRIPTION ;
2. une occurrence du rôle POSSESSEUR ;
3. une ou plusieurs occurrences du rôle PROPRIETE.

SSS1

<Groupe sujet>	(POSSESSEUR)	[(CONTRAINT)]
<Groupe verbal>	(DESCRIPTION)	
<Groupe complément>	(PROPRIETE)	

Exemple

La phrase suivante s'unifie avec le schéma SSS1 :

`A subscriber has a name and an address.`

- `A subscriber` est le <groupe sujet> dont le rôle est POSSESSEUR ;
- `Has` est le <groupe verbal> appartenant à la classe DESCRIPTION ;
- `A name` et `an address` font partie du <groupe complément> dont le rôle est PROPRIETE.

Si le <groupe verbal> est composé d'un <auxiliaire de mode>, alors le <groupe sujet> joue les rôles de PROPRIETE et de CONTRAINT.

Exemple

`Subscribers can have an address.`

- `Subscribers` est le <groupe sujet> dont le rôle est POSSESSEUR & CONTRAINT (la source de contrainte est la présence de l'auxiliaire de mode) ;
- `Can have` est le <groupe verbal> appartenant à la classe DESCRIPTION ;
- `An address` est le <groupe complément> dont le rôle est PROPRIETE.

SSS2

```
<Groupe sujet>          (PROPRIETE)
<Groupe verbal>         (DESCRIPTION)
<Groupe complément>     (PROSSESSEUR)
```

Exemple

La phrase élémentaire suivante s'unifie avec le schéma SSS 2 :

`A name characterizes an author.`

- `A name` est le <groupe sujet> dont le rôle est PROPRIETE ;
- `Characterizes` est le <groupe verbal> appartenant à la classe DESCRIPTION ;
- `An author` est le <groupe complément> dont le rôle est POSSESSEUR.

SSS3

```
<Groupe sujet>          (POSSESSEUR & CONTRAINT)
<Groupe verbal>         (DESCRIPTION)
<Groupe informatif>     (CONTRAINTE)
<Groupe complément>     (PROPRIETE)
```

Exemple

La phrase élémentaire suivante s'unifie avec le schéma SSS 3 :

`A customer has at most five telephon numbers.`

- `A customer` est le <groupe sujet> dont le rôle est POSSESSEUR et CONTRAINT ;
- `Has` est le <groupe verbal> appartenant à la classe DESCRIPTION ;
- `At most five` est le <groupe informatif> appartenant à la classe CONTRAINTE ;
- `Telephon numbers` est le <groupe complément> dont les rôles sont PROPRIETE.

La phrases suivante s'unifie également avec le schéma SSS 3 :

`A customer has at least one surname.`

4.5.1.2 Schémas structuraux identifiants (SSI)

Ils comportent en général les éléments suivants :

1. un verbe de la catégorie IDENTIFICATION ;
2. une occurrence du rôle POSSESSEUR ;
3. une occurrence du rôle IDENTIFIANT.

SSI1

```
<Groupe sujet>          (POSSESSEUR)
<Groupe verbal>         (IDENTIFICATION)
<Groupe complément>     (IDENTIFIANT)
```

Exemple

La phrase élémentaire suivante s'unifie avec le schéma SSI 1 :

`A subscriber is identified by a number.`

- `A subscriber` est le <groupe sujet> dont le rôle est POSSESSEUR ;
- `Is identified by` est le <groupe verbal> appartenant à la classe IDENTIFICATION ;
- `A name` est le <groupe complément> dont le rôle est IDENTIFIANT.

SSI2

```
<Groupe sujet>          (IDENTIFIANT)
<Groupe verbal>         (IDENTIFICATION)
<Groupe complément>     (POSSESSEUR)
```

Exemple

La phrase élémentaire suivante s'unifie avec le schéma SSI 2 :

`A number identifies a subscriber.`

- `A number` est le <groupe sujet> dont le rôle est IDENTIFIANT ;
- `Identifies` est le <groupe verbal> appartenant à la classe IDENTIFICATION ;
- `A subscriber` est le <groupe complément> dont le rôle est POSSESSEUR.

4.5.2 Schémas associatifs : SA

Ces schémas définissent toutes les structures qui associent deux objets du monde du réel. Nous distinguons deux sous-catégories selon qu'il s'agit de schémas décrivant des actions effectuées dans le monde du réel (SAA) ou qu'il s'agit de schémas décrivant une relation de spécialisation d'un objet du réel (SAS).

4.5.2.1 Schémas associatifs d'action (SAA)

Ils comportent en général les éléments suivants :

- Un verbe de la catégorie ACTION.
- Une occurrence du rôle ACTEUR.
- Une occurrence du rôle OBJET.

SAA1

```
<Groupe sujet>          (ACTEUR)          [(CONTRAINT)]
<Groupe verbal>         (ACTION)
<Groupe complément>     (OBJET)
```

Exemple

Les phrases élémentaires suivantes s'unifient avec le schéma SAA 1 :

```
Subscribers borrow some books.
Subscribers borrow books.
```

- `Subscribers` est le <groupe sujet> dont le rôle est ACTEUR ;
- `Borrow` est le <groupe verbal> dont le sens est ACTION ;
- `(Some) books` est le <groupe complément> dont le rôle est OBJET.

Si le <groupe verbal> est composé d'un <auxiliaire de mode>, alors le <groupe sujet> joue les rôles d'ACTEUR et de CONTRAINT.

Exemple

```
Subscribers can borrow some books.
```

- `Subscribers` est le <groupe sujet> dont le rôle est ACTEUR & CONTRAINT (la source de contrainte est la présence de l'auxiliaire de mode) ;
- `Can borrow` est le <groupe verbal> dont le sens est ACTION ;
- `Some books` est le <groupe complément> dont le rôle est OBJET.

SAA2

```
<Groupe sujet>          (ACTEUR & CONTRAINT)
<Groupe verbal>         (ACTION)
<Groupe informatif>     (CONTRAINTE)
<Groupe complément>     (OBJET)
```

Exemple

La phrase élémentaire suivante s'unifie avec le schéma SAA 2 :

`Subscribers borrow at least one book.`

- `Subscribers` est le <groupe sujet> dont le rôle est ACTEUR & CONTRAINT ;
- `Borrow` est le <groupe verbal> dont le sens est ACTION ;
- `At least one` est le <groupe informatif> dont le rôle est CONTRAINTE ;
- `book` est le <groupe complément> dont le rôle est OBJET.

4.5.2.2 Schémas associatifs de spécialisation (SAS)

Ils comportent en général les éléments suivants :

1. un verbe de la catégorie SPECIALISATION ;
2. une occurrence du rôle GENERALISATION ;
3. une mais souvent plusieurs occurrences du rôle SPECIALISATION.

SAS1

```
<Groupe sujet>          (GENERALISATION) [(CONTRAINT)]
<Groupe verbal>         (SPECIALISATION)
<Groupe complément>     (SPECIALISATION)
```

Exemple

La phrase élémentaire suivante s'unifie avec le schéma SAS 1 :

`A subscriber is a professor or a student.`

- `Subscriber` est le <groupe sujet> dont le rôle est GENERALISATION ;
- `Is` est le <groupe verbal> dont le sens est SPECIALISATION ;
- `Professeur, student` sont les éléments du <groupe complément> qui ont un rôle SPECIALISATION.

Si le <groupe verbal> est composé d'un <auxiliaire de mode>, alors le <groupe sujet> joue les rôles de GENERALISATION et de CONTRAINT.

Exemple

`Subscribers must be a professor or a student.`

- `Subscribers` est le <groupe sujet> dont le rôle est GENERALISATON & CONTRAINT (la source de contrainte est la présence de l'auxiliaire de mode) ;
- `Must be` est le <groupe verbal> dont le sens est SPECIALISATION ;
- `Professor, student` sont les éléments du <groupe complément> qui ont un rôle de SPECIALISATION.

SAS2

```
<Groupe sujet>          (GENERALISATION) (CONTRAINT)
<Groupe verbal>         (SPECIALISATION)
<Groupe informatif>     (CONTRAINTE)
<Groupe complément>     (SPECIALISATION)
```

Exemple

La phrase élémentaire suivante s'unifie avec le schéma SAS 2 :

`A subscriber is either a professor or a student.`

- `Subscriber` est le <groupe sujet> dont le rôle est GENERALISATION & CONTRAINT;
- `Is` est le <groupe verbal> dont le sens est SPECIALISATION ;
- `Either` est le <groupe informatif> dont le rôle est CONTRAINT ;
- `Professor, student` sont les éléments du <groupe complément> qui ont un rôle SPECIALISATION.

5. Mise en œuvre de l'approche linguistique

5.1 Introduction

Dans ce chapitre, nous montrons, sans tenir compte de la technique, de quelle manière l'approche linguistique présentée dans le chapitre précédent a été mise en œuvre pour générer un schéma sémantique élémentaire à partir d'un texte structuré de phrases élémentaires.

Le processus est inspiré des différentes études [SABAH, 90b] effectuées dans le domaine de la compréhension des textes en langage naturel. De manière générale, l'analyse et l'interprétation des textes écrits peuvent se diviser en quatre phases :

1. Une **phase morphologique** qui permet de reconnaître les mots sous les différentes formes avec lesquels ils apparaissent dans la phrase.
2. Une **phase lexicale** qui met en correspondance le mot une fois reconnu avec les informations reconnues dans le lexique.
3. Une **phase syntaxique** qui permet de rendre compte de l'agencement des mots les uns par rapport aux autres dans la phrase. Elle permet, par exemple, de reconnaître qu'une phrase est grammaticalement incorrecte et ce indépendamment de tout sens qu'elle peut avoir.
4. Une **phase sémantique** qui fait correspondre des situations du monde réel aux structures reconnues par le niveau syntaxique. Elle permet de comprendre le sens de la phrase.

Notre démarche tient également compte de l'approche linguistique élaborée au chapitre précédent :

1. reconnaissance de schémas types de phrases basée à la fois sur la syntaxe et sur le sens du verbe dans la phrase (cfr. chapitre 3) ;
2. interprétation de ces schémas en termes de concepts du modèle sémantique élémentaire.

Nous avons donc organisé notre processus en un enchaînement de trois phases élémentaires. Cet enchaînement est basé sur les démarches proposées par [PROIX, 89] et [BUCHHOLZ, 96] et illustré dans la figure 5.1.

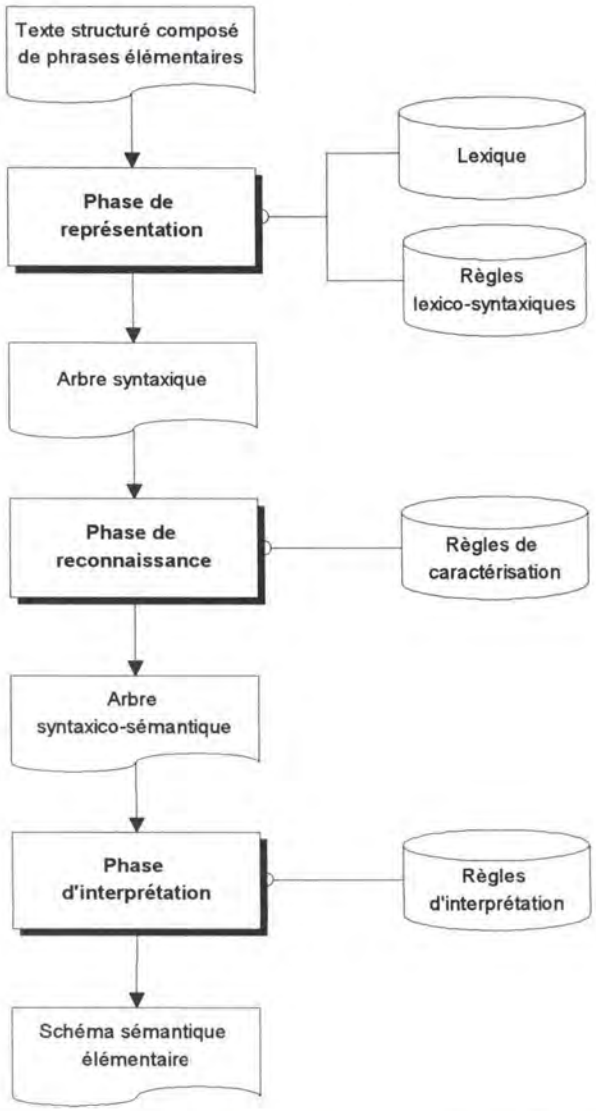


Figure 5.1. Enchaînement des phases du processus de génération d'un schéma sémantique élémentaire

La première phase (la **phase de représentation**) consiste à construire une représentation interne des phrases sous forme d'arbres syntaxiques (cfr. annexe I) mettant en évidence leur structure grammaticale. Cette phase utilise les connaissances morphologique, lexicale et syntaxique citées précédemment.

La **phase de reconnaissance** des schémas caractérise sémantiquement les arbres syntaxiques. Elle détermine le sens du verbe et le rôle de chaque mot. Le produit de cette phase est un arbre syntaxico-sémantique exprimant la sémantique et la syntaxe des phrases.

La **phase d'interprétation** traduit les différents faits linguistiques représentés par les arbres syntaxico-sémantiques en termes de concepts du modèle sémantique élémentaire. Les règles de traduction caractérisent les différentes transformations à exécuter pour obtenir le schéma sémantique élémentaire.

5.2 Phase de représentation

Durant cette phase, les analyses morphologique, lexicale et syntaxique du texte sont réalisées. Le résultat de cette phase est la construction d'un arbre syntaxique mettant en évidence la structure grammaticale de la phrase (cfr. figure 5.2).

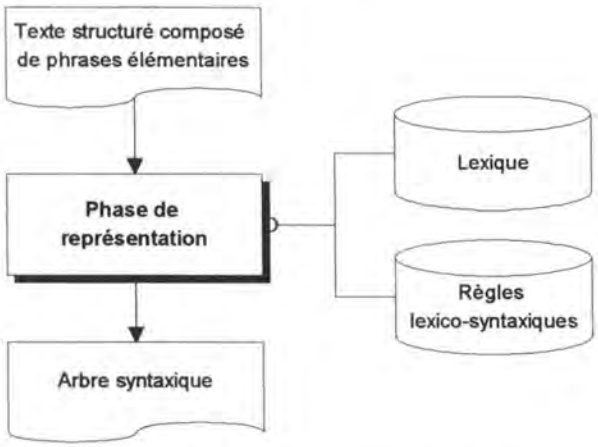


Figure 5.2. Phase de représentation

5.2.1 Analyse morpho-lexicale

Cette analyse a pour buts de :

1. isoler la forme canonique des mots ;
2. reconnaître et déterminer la nature grammaticale de chaque mot ;
3. affecter une valeur sémantique aux verbes.

Ces trois opérations sont effectuées à partir du **lexique**. Le lexique représente la base de connaissance du système sur le vocabulaire anglais. Pour chaque mot qu'il comprend, le lexique mémorise trois types d'information :

1. une information sur l'orthographe du mot : la forme canonique du mot ;
2. une information complémentaire et facultative (si le mot est un verbe, alors l'information complémentaire correspond au substantif du verbe) ;
3. une information sur la nature grammaticale du mot : article, adverbe, conjonction, contrainte, disjonction, mot-clé ou verbe.

Un **mot-clé** est un mot ou groupe de mots (reliés entre eux par un trait d'union) qui limite la signification d'une phrase élémentaire. Il est source d'une contrainte d'intégrité et précise le sens apporté à la phrase. Il existe trois classes de mot-clé :

1. la première classe regroupe les mots sources de contrainte de connectivité minimale (par exemple : `at-least`, `minimum`) ;

2. la deuxième classe regroupe les mots sources de contrainte de connectivité maximale (par exemple, `at-most`, `maximum`) ;
3. la troisième classe regroupe les mots caractérisant les propriétés du type générique (par exemple, `either`).

Les mots-clés sont rangés dans le lexique en fonction de la classe à laquelle ils appartiennent.

Pour les **verbes**, le lexique conserve également la forme primitive du verbe et une information sur la sémantique de ces mots. La forme primitive du verbe est le mot utilisé pour dénommer le TA (cfr. paragraphe 5.4.1). Il est souvent assimilé au **substantif** du verbe. L'information sémantique correspond aux **sens** du verbe, sens qui seront définis dans le paragraphe 5.31.

La figure 5.3. présente la structure du lexique selon le formalisme Entité-Association.

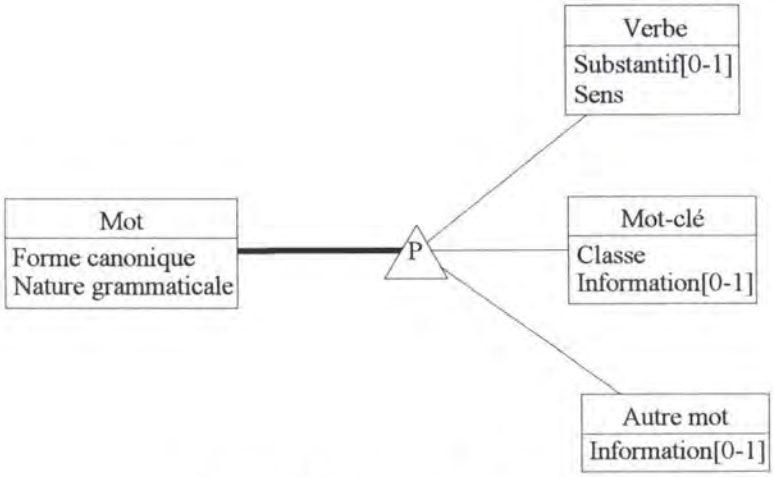


Figure 5.3. Structure du lexique

Afin de réduire la taille du lexique, nous avons choisi :

1. **De ne mémoriser que la forme canonique des mots.** Pour isoler la forme canonique des mots, nous avons mis en œuvre un mécanisme d'analyse pour reconnaître les formes canoniques à partir des formes fléchies (cfr. chapitre 6, paragraphe 6.3.2).
2. **De ne pas mémoriser obligatoirement les verbes d'action** qui correspondent au vocabulaire particulier du domaine d'application (cfr. chapitre 4, paragraphe 4.4.2). Ils sont ajoutés dans le lexique lorsque le domaine d'application auquel ils appartiennent est rencontré.
3. **De ne pas mémoriser les noms** qui correspondent au vocabulaire particulier du domaine d'application.

Le lexique ne comporte donc que les mots et un sous-ensemble des verbes utilisés. Par conséquent, au cours de l'analyse morpho-lexicale, tous les mots qui ne sont pas dans le lexique peuvent être des verbes ou des noms. L'analyse morpho-lexicale n'est donc pas entièrement déterministe.

Considérons, par exemple, la phrase suivante et supposons que le verbe `borrow` soit absent du lexique :

```
A subscriber borrows at-most five book.
```

L'analyse morpho-lexicale ne peut pas déterminer la nature grammaticale de `borrows` : elle ne peut en effet déterminer la nature grammaticale d'un mot absent du lexique. Pour déterminer sa nature grammaticale, il est nécessaire de connaître la place du mot dans la phrase. Elle doit donc être levée par un mécanisme de retour en arrière dans la phase d'analyse syntaxique : chaque fois qu'un mot de nature grammaticale indéterminée est reconnu comme verbe, il est ajouté dans le lexique.

Le résultat de cette phase est la production de la **forme canonique** et **grammaticale** des mots.

5.2.2 Analyse syntaxique

Cette analyse permet de vérifier d'une part que les phrases appartiennent bien au langage accepté par l'outil et d'autre part de construire les **arbres syntaxiques** qui représentent leurs structures grammaticales.

Les règles syntaxiques qui régissent cette analyse sont basées sur l'emploi de la **grammaire générative de Chomsky** ([CHOMSKY, 65]). Elle définit de manière formelle la grammaire autorisée. Une telle grammaire est composée d'un vocabulaire terminal, d'un vocabulaire non terminal et d'un ensemble de règles de production qui définissent l'ensemble des structures grammaticales autorisées. Le vocabulaire terminal correspond au contenu du lexique. Le vocabulaire non terminal est l'ensemble des expressions correspondant aux fonctions grammaticales de mots (par exemple, groupe sujet, groupe verbal, etc.).

Les règles de production ont le format suivant :

$$A \rightarrow a$$

où **A** est un élément du vocabulaire non terminal et **a** est une suite d'éléments des vocabulaires terminal et non terminal.

Vocabulaire non terminal :

(phrase, nom, groupe sujet, groupe verbal, groupe informatif, groupe complément, verbe inconnu, article, auxiliaire de mode, auxiliaire être, mot-clé, séparateur logique, préposition, verbe du lexique)

Règles de production :

```
<Phrase> →      <groupe sujet>
              | <groupe sujet> <groupe verbal> [<groupe informatif>]
                <groupe complément>
```

```
<groupe sujet>            → <groupe nominal simple>

<groupe complément>       → <groupe nominal>
                            [{<groupe nominal>}
                              <séparateur logique>
                               <groupe nominal> ]

<groupe nominal simple>   → [<article>] <nom>

<groupe nominal>          → [<entier>|<article>] <nom>

<groupe informatif>       → <mot-clé>

<groupe verbal>           →  <auxiliaire être>
                             [<auxiliaire de mode>]
                             <verbe> [<préposition>|<adverbe>]

<verbe>                   → <verbe du lexique>|<verbe inconnu>
```

Exemple

La grammaire générative permet de construire l'arbre syntaxique de la phrase suivante : « `A subscriber has a name and an address.` » L'arbre syntaxique de cette phrase est présenté à titre d'illustration dans l'annexe I.

Cette grammaire ne décrit qu'une grammaire simplifiée de l'anglais. Cependant, cette grammaire est suffisante pour formuler des phrases élémentaires dans l'optique d'une description d'un domaine d'application.

Le principales règles grammaticales peuvent être résumées comme suit :

1. La construction de base est **une phrase élémentaire, déclarative et affirmative**.
2. Une phrase élémentaire commence par une majuscule et se termine par un point.
3. Le <groupe complément> est un groupe de mots composé d'un ou plusieurs <groupe nominal>. Les deux derniers <groupe nominal> du <groupe complément> sont reliés par `AND` ou `OR`. Les premiers sont reliés entre eux par une virgule.
4. Le <verbe conjugué> est la forme fléchie du verbe mémorisée dans le lexique si le verbe est dans le lexique.
5. L'inversion du groupe sujet par rapport au groupe verbal n'est pas autorisé.

Le produit de l'analyse syntaxique est un arbre syntaxique représentant la structure grammaticale de la phrase. La stratégie que nous employons consiste à :

1. Décomposer la phrase en un groupe sujet, en un groupe verbal, en un groupe complément et en un groupe informatif. Le groupe sujet est assimilé au premier groupe de mots de la phrase.
2. Appliquer les règles de production de la grammaire aux différents groupes de la phrase.
3. Si un mot est reconnu comme <verbe inconnu>, l'ajouter dans le lexique en tant que verbe d'action.

Le résultat de cette analyse est **un ensemble de faits organisés sous le forme d'un arbre syntaxique.**

5.3 Phase de reconnaissance

Durant cette phase, le système détermine les rôles des différents mots inclus dans la phrase. Les règles de caractérisation des rôles ont pour but de caractériser **sémantiquement** (c'est-à-dire de définir les rôles) chaque fait linguistique. Le résultat de cette phase est la construction d'un arbre syntaxico-sémantique (cfr. figure 5.4).

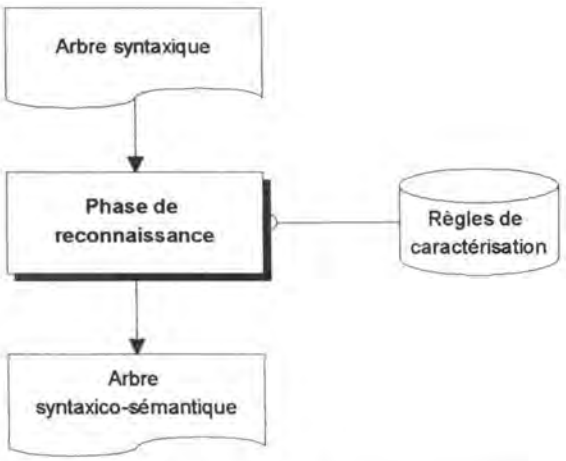


Figure 5.4. Phase de reconnaissance

5.3.1 Classes sémantiques des verbes

Comme nous l'avons exposé précédemment, le lexique comporte une partie des verbes utilisés dans les textes. Ces verbes y sont regroupés en quatre grandes classes de **sens théorique** qui sont dérivées des classes sémantiques de verbe présentées au chapitre quatre :

1. la classe **POS** correspondant à la classe sémantique **description**;
2. la classe **IDENT** correspondant à la classe sémantique **identification**;
3. la classe **SPEC** correspondant à la classe sémantique **spécification**;
4. la classe **ACTION** correspondant à la classe sémantique **action.**

5.3.1.1 Verbes de type POS

La classe **POS** (qui correspond à la classe sémantique **description**) regroupe les verbes qui expriment une relation simple entre deux objets. A ces verbes sont associés les rôles de **POSSESSEUR** et de **PROPRIETE**.

La classe **POS** renferme les sous-classes **POS-SUJ** et **POS-COMPL**. La classe **POS-SUJ** regroupe les verbes pour lesquels le rôle POSSESSEUR est assumé par le groupe sujet.

Exemple

Le verbe « `have` » dans la proposition « `a subscriber has a name` ». Le rôle POSSESSEUR y est tenu par le groupe sujet « `a subscriber` » et le rôle PROPRIETE y est tenu pas le groupe complément « `a name` ».

La classe **POS-COMPL** regroupe les verbes pour lesquels le rôle POSSESSEUR est assumé par le groupe complément.

Exemple

Le verbe « `characterize` » dans le proposition « `an address characterizes a subscriber` ». Le rôle POSSESSEUR y est tenu par le groupe complément « `a subscriber` » et le rôle PROPRIETE par le groupe sujet « `an address` ».

5.3.1.2 Verbes de type IDENT

La classe **IDENT** regroupe les verbes qui expriment une relation d'identifiant entre deux objets. A ces verbes sont associés les rôles de **POSSESSEUR** et d'**IDENTIFIANT**.

La classe **IDENT** renferme les sous-classes **IDENT-SUJ** et **IDENT-COMPL**. La classe **IDENT-SUJ** regroupe les verbes pour lesquels le rôle POSSESSEUR est assumé par le groupe sujet.

La classe **IDENT-COMPL** regroupe les verbes pour lesquels le rôle POSSESSEUR est assumé par le groupe complément.

Exemple

Le verbe « `identifies` » dans la proposition « `a number identifies a subscriber` ». Le rôle POSSESSEUR y est tenu par le groupe complément « `a subscriber` » et le rôle IDENTIFIANT par le groupe sujet « `a number` ».

5.3.1.3 Verbes de type SPECIALISATION

La classe **SPECIALISATION** regroupe les verbes exprimant une relation de spécialisation (relation `is-a`). A chacun de ces verbes est associé les rôles de SPECIALISATION et de GENERALISATION ; le rôle GENERALISATION étant tenu par l'objet qui est le sur-type.

Exemple

Le verbe « `be` » dans la proposition « `a road can be a motorway or a national road` ». Le rôle SPECIALISATION y est tenu par « `a motorway` » et par « `a national road` ». Le rôle GENERALISATION y est tenu par le groupe sujet « `road` ».

5.3.1.4 Verbes de type ACTION

La classe **ACTION** regroupe les verbes exprimant une action. Il s'agit de verbes étroitement liés au domaine d'application et non nécessairement présents dans le lexique. A chacun de ces verbes est associé les rôles d'ACTEUR et d'OBJET. Le rôle d'OBJET étant tenu par l'objet qui est l'objet de l'action.

Exemple

Le verbe « `borrows` » dans le proposition « `a subscriber borrows a book` ». Le rôle ACTEUR y est tenu par le groupe sujet « `a subscriber` » et le rôle OBJET par le groupe complément « `a book` ».

5.3.2 Règles de détermination des rôles

Les règles de détermination des rôles au sein d'une phrase doivent :

1. être capable d'établir le **sens réel** d'un verbe ; ceci se fait en utilisant le **sens théorique** du verbe (c'est-à-dire le sens de verbe défini au paragraphe 5.3.1) et la **voie** grammaticale du groupe verbal (voie passive ou active) ;
2. être capable de déterminer les rôles de chaque élément de la phrase.

Ces règles sont actuellement au nombre de neuf. Nous en donnons ci-dessous les principes.

```
REGLE-ROLE 1

SI VOIE(groupe verbal)= PASSIVE  ALORS
      SI SENS-THEORIQUE(verbe conjugué) = POS-SUJ
      ALORS SENS-REEL(verbe conjugué) = POS-COMPL

      SI SENS-THEORIQUE(verbe conjugué) = POS-COMPL
      ALORS SENS-REEL(verbe conjugué) = POS-SUJ

      SI SENS-THEORIQUE(verbe conjugué) = INDENT-SUJ
      ALORS SENS-REEL(verbe conjugué) = IDENT-COMPL

      SI SENS-THEORIQUE(verbe conjugué) = INDENT-COMPL
      ALORS SENS-REEL(verbe conjugué) = IDENT-SUJ

      SI SENS-THEORIQUE(verbe conjugué) = SPECIALISATION
      ALORS SENS-REEL(verbe conjugué) = SPECIALISATION

      SI SENS-THEORIQUE(verbe conjugué) = ACTION
      ALORS SENS-REEL(verbe conjugué) = ACTION
```

REGLE-ROLE 2

```
SI SENS-REEL(verbe conjugué) = POS-SUJ
ALORS
      ROLE(groupe sujet) = POSSESSEUR
      ROLE(groupe complément) = PROPRIETE
```

Exemple

Les règles REGLE-ROLE 1 et REGLE-ROLE 2 appliquées à la proposition suivante : « `A book is referenced by a title and a date.` » permettent de déterminer que :

- le <groupe verbal> a pour sens réel : POS-SUJ ;
- l'élément syntaxique `book` joue le rôle de POSSESSEUR ;
- les éléments syntaxiques `title` et `date` jouent le rôle de PROPRIETE.

REGLE-ROLE 3

```
SI SENS-REEL(verbe conjugué) = POS-COMPL
ALORS
      ROLE(groupe sujet) = PROPRIETE
      ROLE(groupe complément) = POSSESSEUR
```

REGLE-ROLE 4

```
SI SENS-REEL(verbe conjugué) = INDENT-SUJ
ALORS
      ROLE(groupe sujet) = POSSESSEUR
      ROLE(groupe complément) = IDENTIFIANT
```

REGLE-ROLE 5

```
SI SENS-REEL(verbe conjugué) = IDENT-COMPL
ALORS
      ROLE(groupe sujet) = IDENTIFIANT
      ROLE(groupe complément) = POSSESSEUR
```

REGLE-ROLE 6

```
SI SENS-REEL(verbe conjugué) = ACTION
ALORS
      ROLE(groupe sujet) = ACTEUR
      ROLE(groupe complément) = OBJET
```

REGLE-ROLE 7

```
SI SENS-REEL(verbe conjugué) = SPECIALISATION
ALORS
      ROLE(groupe sujet) = GENERALISATION
      ROLE(groupe informatif) = SPECIALISATION
```

```
REGLE-ROLE 8

SI EXISTE(groupe informatif)
ALORS
      ROLE(groupe sujet) = CONTRAINT
      ROLE(groupe informatif) = CONTRAINTE
```

```
REGLE-ROLE 9

SI EXISTE(auxiliaire de mode)
ALORS
      ROLE(groupe sujet) = CONTRAINT
```

5.4 Phase d'interprétation

Durant cette phase, le système utilise des règles d'interprétation pour traduire l'arbre syntaxico-sémantique en terme de concepts du modèle sémantique élémentaire. La construction du schéma sémantique élémentaire est effectué à partir de règles (cfr. figure 5.5).

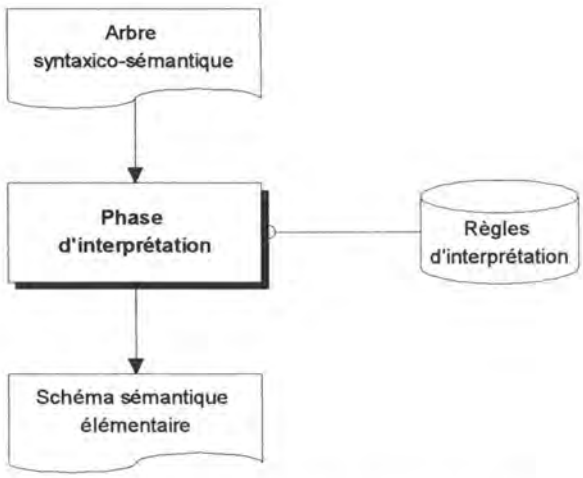


Figure 5.5. Phase d'interprétation

Les règles d'interprétation sont basées sur le principe général de faire correspondre, en fonction du schéma de la phrase, un concept du schéma sémantique élémentaire à un rôle sémantique. Les règles d'interprétation aboutissent à la construction d'un schéma sémantique élémentaire.

Nous distinguons trois classes de règles d'interprétation :

1. les règles d'interprétation des schémas ;
2. les règles de détermination des cardinalités des T.A ;
3. les règles déterminant les propriétés du type générique.

5.4.1 Interprétation des schémas

Les règles d'interprétation des schémas peuvent être classées en fonction du schéma qu'elles interprètent. Nous distinguons :

1. les règles interprétant les schémas SSS1, SSS2 et SSS3 ;
2. les règles interprétant les schémas SSI1 et SSI2 ;
3. les règles interprétant les schémas SAA1 et SAA2 ;
4. les règles interprétant les schémas SAS1 et SAS2.

5.4.1.1 Règles interprétant les schémas SSS1, SSS2 et SSS3

Ces règles permettent d'interpréter les propositions qui s'unifient avec les schémas SSS1, SSS2 et SSS3. Elles aboutissent à la construction de TE, de TA et d'attributs.

```
REGLE-INTERPRETATION-SSS1-SSS2-SSS3

IF SENS-REEL(forme-verbale(phrase)) = POS-SUJ OR POS-COMPL

THEN  Construire avec le groupe nominal associé au rôle POSSESSEUR un
      TE et attribuer au TE le nom du groupe nominal Si le TE n'existe pas
      encore.

      Construire avec chaque groupe nominal associé au rôle PROPRIETE un
      TE. Attribuer à chaque TE le nom du groupe nominal.
      Insérer dans chaque TE associé au rôle PROPRIETE un attribut
      identifiant de même nom que le TE qu'il caractérise.
      Relier ces TE par des TA dont l'origine est le TE associé au rôle
      POSSESSEUR et dont les extrémités sont les TE associés au rôle
      PROPRIETE.

      Positionner les cardinalités des rôles des TA
            (TE Origine = TE associé au rôle POSSESSEUR ;
             TE Destination = TE associé(s) au rôle PROPRIETE)
```

Exemple

Soit la phrase « A subscriber has a name AND an address ». L'analyse lexicale et la phase de caractérisation sémantique conduisent aux faits suivants (seuls les faits utilisés par la règle d'interprétation sont présentés) :

```
(subscriber, POSSESSEUR)
(have, POS-SUJ)
(name, PROPRIETE)
(address, PROPRIETE)
```

L'application de la règle d'interprétation décrite ci-dessus et des règles de détermination des cardinalités (voir paragraphe suivant) aboutit à la construction du schéma sémantique élémentaire de la figure 5.6.

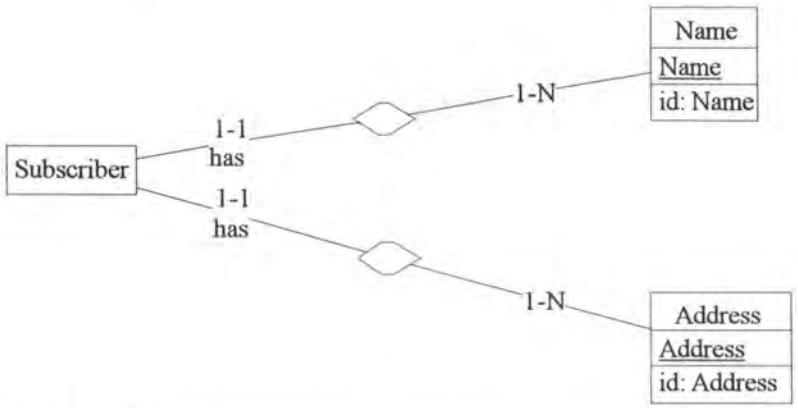


Figure 5.6. Schéma sémantique élémentaire obtenu à partir des règles d'interprétation SSS1, SSS2 et SSS3

5.4.1.2 Règles interprétant les schémas SSI1 et SSI2

Ces règles permettent d'interpréter les propositions qui s'unifient avec les schémas SSI1 et SSI2. Elles aboutissent à la construction de TE, de TA et d'attributs identifiants.

```
REGLE-INTERPRETATION-SSI1-SSI2

IF SENS-REEL(forme-verbale(phrase)) = IDENT-SUJ OR IDENT-COMP

THEN  Construire avec le groupe nominal associé au rôle POSSESSEUR un TE
      si le TE n'existe pas encore.
      Attribuer au TE le nom du groupe nominal.

      Construire avec chaque groupe nominal associé au rôle IDENTIFIANT un
      TE.
      Attribuer à chaque TE le nom du groupe nominal.
      Insérer dans chaque TE associé au rôle IDENTIFIANT un attribut
      identifiant de même nom que le TE qu'il caractérise.

      Relier ces TE par des TA dont l'origine est le TE associé au rôle
      POSSESSEUR et dont les extrémités sont les TE associés au rôle
      IDENTIFIANT.

      Positionner les cardinalités des rôles des TA
            (TE Origine = TE associé au rôle POSSESSEUR ;
             TE Destination = TE associé(s) au rôle IDENTIFIANT)
```

Exemple

Soit la phrase « A subscriber is identified by a name ». L'application de la règle d'interprétation décrite ci-dessus et des règles de détermination des cardinalités (voir paragraphe suivant) aboutit à la construction du schéma sémantique élémentaire représenté à la figure 5.7.

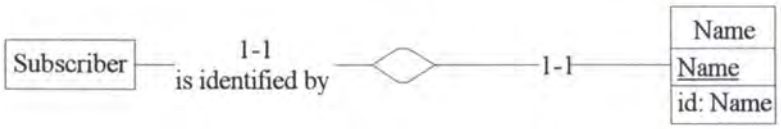


Figure 5.7. Schéma sémantique élémentaire obtenu à partir des règles d'interprétation SSI1 et SSI2

5.4.1.3 Règles interprétant les schémas SAA1 et SAA2

Ces règles permettent d'interpréter les propositions qui s'unifient avec les schémas SAA1 et SAA2. Elles aboutissent à la construction de TE, de TA.

```
REGLE-INTERPRETATION-SAA1-SAA2

IF SENS-REEL(forme-verbale(phrase)) = ACTION

THEN  Construire avec le groupe nominal associé au rôle ACTEUR un TE si le
      TE n'existe pas encore.
      Attribuer au TE le nom du groupe nominal.

      Construire avec chaque groupe nominal associé au rôle OBJET un TE si
      le TE n'existe pas encore.
      Attribuer à chaque TE le nom du groupe nominal.

      Relier ces TE par des TA dont l'origine est le TE associé au rôle
      ACTEUR et dont les extrémités sont les TE associés au rôle OBJET.

      Attribuer au TA le substantif attaché au verbe ou, à
      défaut, la forme canonique du verbe.
      Attribuer au rôle joué par le TE ACTEUR le verbe conjugué.

      Positionner les cardinalités des rôles des TA
            (TE Origine = TE associé au rôle ACTEUR ;
             TE Destination = TE associé(s) au rôle OBJET)
```

Exemple

Soit la phrase « `Subscribers borrow at-most 5 books` ». L'application de la règle d'interprétation décrite ci-dessus et des règles de détermination des cardinalités (voir paragraphe suivant) aboutit à la construction du schéma sémantique élémentaire de la figure 5.8.

Figure 5.8. Schéma sémantique élémentaire obtenu à partir des règles d'interprétation SAA1 et SAA2

5.4.1.4 Règles interprétant les schémas SAS1 et SAS2

Ces règles permettent d'interpréter les propositions qui s'unifient avec les schémas SAS1 et SAS2. Elles aboutissent à la construction de TE et de relations IS-A.

```
REGLE-INTERPRETATION-SAS1-SAS2

IF SENS-REEL(forme-verbale(phrase)) = SPECIALISATION

THEN  Construire avec le groupe nominal associé au rôle GENERALISATION un
      TE si le TE n'existe pas encore.
      Attribuer au TE le nom du groupe nominal.

      Construire avec chaque groupe nominal associé au rôle
      SPECIALISATION un TE si le TE n'existe pas encore.
      Attribuer à chaque TE le nom du groupe nominal.

      Relier ces TE par des relations is-a dont l'origine est le TE associé
      au rôle GENERALISATION et dont les extrémités sont les TE associés
      au rôle SPECIALISATION.

      Déterminer les propriétés du type générique (= TE associé
      au rôle GENERALISATION).
```

Exemple

Soit la phrase « An employee can be a woman or a man ». L'application de la règle d'interprétation décrite ci-dessus et des règles de détermination des cardinalités (voir paragraphe suivant) aboutit à la construction du schéma sémantique élémentaire de la figure 5.9.

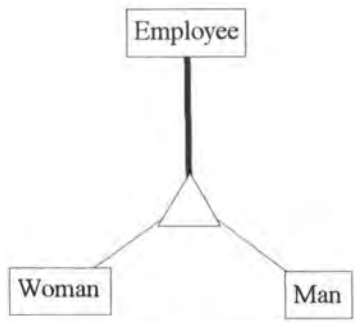


Figure 5.9. Schéma sémantique élémentaire obtenu à partir des règles d'interprétation SAS1 et SAS2

5.4.2 Règles de détermination des cardinalités

Les règles de détermination des cardinalités sont également classées en fonction du schéma qu'elles interprètent :

1. les règles élémentaires, appliquées à tous les schémas structuraux ;
2. les règles de base, appliquées à tous les schémas structuraux et associatifs d'action ;
3. les règles interprétant les schémas structuraux simples et identifiants (schémas SSS1, SSS2, SSS3, SSI1 et SSI2) ;
4. les règles interprétant les schémas associatifs d'action (schémas SAA1 et SAA2).

L'application de ces règles détermine la valeur des cardinalités attachées à chaque rôle. Pour obtenir ce résultat, ces règles produisent 2 vecteurs (CARD-ORIGINE, CARD-DESTINATION) à deux dimensions attachés au TA. Le premier vecteur contient les cardinalités minimale et maximale du rôle joué par un TE origine. Quant au second vecteur, il contient les cardinalités minimale et maximale du rôle joué par le TE destination.

5.4.2.1 Règles élémentaires

Les règles élémentaires s'appliquent aux schémas structuraux simples et identifiants. Elles permettent de fixer la valeur des cardinalités attachées au rôle joué par le groupe nominal ayant pour rôle PROPRIETE ou IDENTIFIANT.

La première règle détermine les cardinalités attachées au rôle joué par un groupe nominal ayant pour rôle PROPRIETE.

```
REGLE-ELEMENTAIRE-1

IF SENS-REEL(forme-verbale(phrase)) = POS-SUJ OR POS-COMPL

THEN  CARD-DESTINATION = [1,N]
```

Exemple

Soit une phrase qui s'unifie avec SSS1 : « A subscriber has a name ».
L'application de la règle interprétant un schéma structurel simple et de la règle décrite ci-dessus aboutit à la construction du schéma sémantique élémentaire de la figure 5.10 (seules les cardinalités déterminées par la REGLE-ELEMENTAIRE-1 sont présentées).

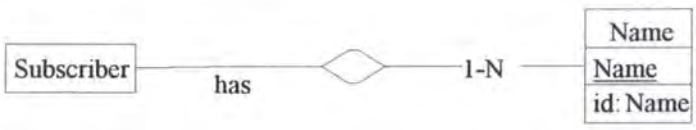


Figure 5.10. Résultat de l'application de la règle interprétant un schéma structurel simple et de la règle REGLE-ELEMENTAIRE-1

La seconde règle détermine les cardinalités attachées au rôle joué par un groupe nominal ayant pour rôle IDENTIFIANT.

```
REGLE-ELEMENTAIRE-2

IF SENS-REEL(forme-verbale(phrase)) = IDENT-SUJ OR IDENT-COMPL

THEN  CARD-DESTINATION = [1,1]
```

Exemple

Soit une phrase qui s'unifie avec SSI1 : « A subscriber is identified by a name ».
L'application de la règle interprétant un schéma structurel identifiant et de la règle décrite ci-dessus aboutit à la construction du schéma sémantique élémentaire de la figure 5.11 (seules les cardinalités déterminées par la REGLE-ELEMENTAIRE-2 sont présentées).

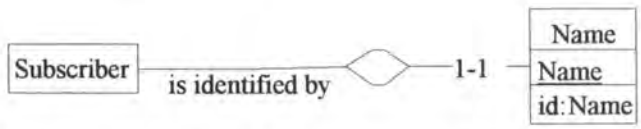


Figure 5.11. Résultat de l'application de la règle interprétant un schéma structurel simple et de la règle REGLE-ELEMENTAIRE-2

5.4.2.2 Règles de base

Les **règles de base** se basent sur la présence ou l'absence d'auxiliaire de mode et/ou de groupe informatif. Elles établissent une première correspondance entre la valeur de la cardinalité attachée au rôle joué par le groupe nominal ayant pour rôle CONTRAINT. L'application de ces règles peut laisser des valeurs de **cardinalité indéterminée**. Ces valeurs seront complétées par les règles spécifiques aux schémas.

5.4.2.2.1 Cardinalités et auxiliaire de mode

On peut établir une correspondance entre la cardinalité minimale d'un rôle et le type d'auxiliaire de mode du groupe verbal. Pour établir cette correspondance, le système utilise des règles basées sur le type d'auxiliaire de mode. Ces règles sont résumées dans le tableau suivant. Le tableau 5.1 donne la valeur de la **cardinalité minimale** de CARD-ORIGINE. Le point d'exclamation désigne le cas où la phrase ne contient pas d'auxiliaire de mode. Le point d'interrogation désigne le cas où les règles de base sont incapables de déterminer la valeur.

Auxiliaire de mode		Cardinalité
CAN	⇔	[0-?]
MUST	⇔	[1-?]
!	⇔	[?-?]

Tableau 5.1. Correspondance entre auxiliaire de mode et cardinalité

Exemple

Soit la phrase : « A subscriber can borrow a book ». Cette phrase s'unifie avec le schéma SAA1 et le rôle CONTRAINT est joué par subscriber (cfr. paragraphe 5.3.2).
L'application de la règle interprétant un schéma associatif d'action et des règles décrites ci-dessus aboutit à la construction du schéma sémantique élémentaire de la figure 5.12 (seules les cardinalités déterminées à partir du tableau 5.1 sont présentées).

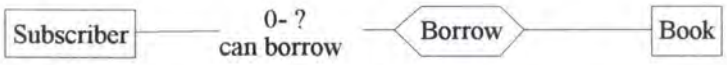


Figure 5.12. Résultat de l'application de la règle interprétant un schéma structurel simple et des règles de correspondance entre auxiliaire de mode et cardinalité

5.4.2.2.2 Cardinalité et groupe informatif

La présence d'un groupe informatif permet au système de déterminer la **cardinalité minimale** ou **maximale** de CARD-ORIGINE selon qu'il s'agisse d'un groupe informatif de type minimum ou de type maximum. Les règles de détermination des cardinalités déterminent la correspondance entre groupes informatifs et valeurs de cardinalité. Le tableau 5.2 illustre cette correspondance. Le point d'exclamation désigne le cas où la phrase ne contient pas de groupe informatif ou d'entier. Le point d'interrogation désigne le cas où les règles de base sont incapables de déterminer la valeur.

Groupe informatif	Entier		Cardinalité	Validité
AT-LEAST	i	⇔	[i-?]	Si i ≤ 1
MINIMUM	i	⇔	[i-?]	Si i ≤ 1
AT-MOST	i	⇔	[?-i]	Si i ≥ 1
MAXIMUM	i	⇔	[?-i]	Si i ≥ 1
ONLY	i	⇔	[?-i]	Si i ≥ 1
!	i	⇔	[?-i]	Si i ≥ 1
!	!	⇔	[?-?]	

Tableau 5.2. Correspondance entre groupe informatif et cardinalité

Exemple

Soit la phrase : « A subscriber has at-most 5 phone-numbers ». Cette phrase s'unifie avec le schéma SSS3 et le rôle CONTRAINT est joué par subscriber (cfr. paragraphe 5.3.2).
L'application de la règle interprétant un schéma structural simple et des règles décrites ci-dessus aboutit à la construction du schéma sémantique élémentaire de la figure 5.13 (seules les cardinalités déterminées à partir du tableau 5.2 sont présentées).

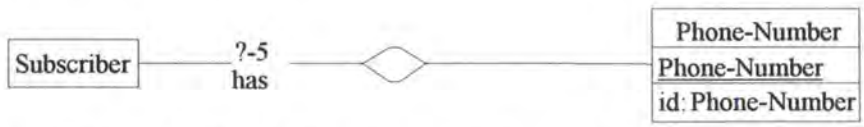


Figure 5.13. Résultat de l'application de la règle interprétant un schéma structurel simple et des règles de correspondance entre cardinalité et groupe informatif

5.4.2.3 Règle interprétant les schémas structuraux

Ces règles permettent d'interpréter les cardinalités des propositions qui s'unifient avec les schémas structuraux. Elles déterminent les cardinalités indéterminées par les règles de base.

La première règle détermine les cardinalités attachées au rôle joué par un groupe nominal ayant pour rôle POSSESSEUR et qui n'ont pas été déterminées par les règles de base. Elle fixe les cardinalités par défaut et s'applique à tous les schémas structuraux.

```
REGLE-CARDINALITE-STRUCTURE-1

IF CARD-ORIGINE(MINIMUM)= ?
THEN
      CARD-ORIGINE(MINIMUM) = 1

IF CARD-ORIGINE(MAXIMUM)= ?
THEN
      CARD-ORIGINE(MAXIMUM) = 1
```

Exemple

Dans l'exemple précédent (`A subscriber has at-most 5 phone-numbers`), les règles de base ont conduit à déterminer la cardinalité maximale du rôle joué par `subscriber` mais pas sa cardinalité minimale. La valeur de cette dernière est obtenue par application de la règle décrite ci-dessus. Elle permet de compléter le schéma de la figure 5.13.

Figure 5.14. Résultat de l'application de la règle REGLE-CARDINALITE-STRUCTURE-1 au schéma de la figure 5.13

La seconde règle rend compte du cas où aucune information sur les cardinalités n'est disponible. Il s'agit du cas où la phrase ne contient pas de groupe nominal ayant pour rôle CONTRAINT.

```
REGLE-CARDINALITE-STRUCTURE-2

IF NOT(EXIST(CONTRAINT))
THEN
      CARD-ORIGINE = [1,1]
```

Exemple

Soit la phrase : « `A subscriber is characterized by a name` ». Cette phrase s'unifie avec le schéma SSS1 et le rôle CONTRAINT n'est joué par aucun groupe de mots.
Les règles de base sont donc incapables de fixer les valeurs de cardinalités associées au rôle joué par `subscriber`. L'application de la règle décrite ci-dessus attribue les valeurs par défaut de ces cardinalités.

5.4.2.4 Règles interprétant les schémas associatifs d'action

Ces règles permettent d'interpréter les cardinalités des propositions qui s'unifient avec les schémas associatifs d'action. Elles déterminent les cardinalités indéterminées par les règles de base.

La première règle fixe par défaut les cardinalités attachées au rôle joué par le groupe nominal ayant pour rôle OBJET.

```
REGLE-CARDINALITE-ACTION-1

      CARD-DESTINATION = [0-N]
```

Exemple

Soit l'exemple : « A subscriber borrows a book ».
L'application de la règle interprétant un schéma structurel simple et de la règle décrite ci-dessus aboutit à la construction du schéma sémantique élémentaire de la figure 5.15 (seules les cardinalités déterminées par la REGLE-CARDINALITE-ACTION-1 sont présentées) :

Figure 5.15. Résultat de l'application de la règle interprétant un schéma structurel simple et de la règle REGLE-CARDINALITE-ACTION-1

La deuxième règle rend compte du cas où aucune information sur les cardinalités n'est disponible. Il s'agit du cas où la phrase ne contient pas de groupe nominal ayant pour rôle CONTRAINT. Cette règle examine le déterminant du groupe complément pour fixer la valeur de CARD-ORIGINE.

```
REGLE-CARDINALITE-ACTION-2

IF NOT(EXIST(CONTRAINT))
THEN
      CARD-ORIGINE = [0-N]
```

Exemple

Dans l'exemple précédent (A subscriber borrows a book), aucun groupe de mots ne joue le rôle CONTRAINT. Les règles de base sont donc incapables de fixer les valeurs des cardinalités associées au rôle joué par subscriber. L'application de la règle décrite ci-dessus attribue les valeurs par défaut de ces cardinalités.

La troisième règle détermine les cardinalités CARD-ORIGINE qui n'ont pas été déterminées par les règles précédentes. Elle fixe les cardinalités par défaut et s'applique à tous les schémas associatifs d'action.

```
REGLE-CARDINALITE-ACTION-3

IF CARD-ORIGINE(MINIMUM)= ?
THEN
      CARD-ORIGINE(MINIMUM) = 0

IF CARD-ORIGINE(MAXIMUM)= ?
THEN
      CARD-ORIGINE(MAXIMUM) = N
```

5.4.3 Règles déterminant les propriétés du type générique

Les **règles déterminant les propriétés du type générique** se basent sur la présence ou l'absence d'auxiliaire de mode et/ou de groupe informatif appartenant à la troisième classe (EITHER, par exemple). L'application de ces règles permet de déterminer si le type générique est disjoint et/ou total. Ces règles sont résumées dans le tableau 5.3.

Auxiliaire de mode	Groupe informatif		Type générique
!	!	⇔	/
!	EITHER	⇔	disjoint
CAN	!	⇔	/
CAN	EITHER	⇔	disjoint
MUST	!	⇔	total
MUST	EITHER	⇔	disjoint & total

Tableau 5.3. Correspondance entre auxiliaire de mode, groupe informatif et type de relation de sous-typage

Exemple

Soit l'exemple : « A subscriber must be either a professor or a student ».
L'application de la règle interprétant un schéma associatif de spécialisation et des règles décrites ci-dessus aboutit à la construction du schéma sémantique élémentaire de la figure 5.16.

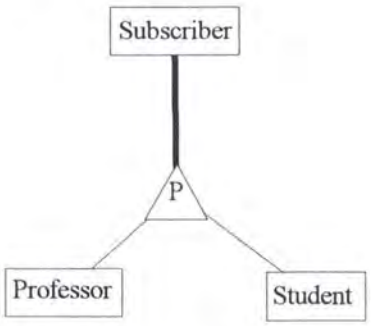


Figure 5.16. Résultat de l'application de la règle interprétant un schéma structurel simple et des règles de détermination des propriétés du type générique

5.5 Exemple

Pour illustrer les différentes phases de mise en œuvre de l'approche linguistique, nous analyserons, phase par phase, les trois phrases suivantes :

```
A subscriber has a name and an address.
A book is identified by a number.
A subscriber borrows at-most 5 books.
```

Le verbe `borrow` est un verbe qui correspond au vocabulaire particulier du domaine d'application (gestion d'une bibliothèque). Nous le considérons absent du lexique.

5.5.1 Phase de représentation

5.5.1.1 Analyse morpho-lexicale

Le résultat (simplifié) de l'analyse lexico-morphologique est le suivant :

Pour la première phrase :
(A, déterminant, /) ; (subscriber, ?, /) ; (have, verbe, description) ; (A, déterminant, /) ; (name, ?, /) ; (and, conjonction, /) ; (An, déterminant, /) ; (address, ?, /)

Pour la seconde phrase :
(A, déterminant, /) ; (book, ?, /) ; (be, verbe, auxiliaire être); (identifie, verbe, identifiant) ; (a, déterminant,/) ; (number, ?, /)

Pour la troisième phrase :
(A, déterminant, /) ; (subscriber, ?, /) ; (borrow, ?, ?) ; (at-most, information, maximum) ; (5, entier, /) ; (book, ?, /)

L'analyse morpho-lexicale n'a pas déterminé la nature grammaticale du mot `borrow` car celui-ci est absent du lexique.

5.5.1.2 Analyse syntaxique

L'analyse syntaxique produit un arbre syntaxique mettant en évidence la structure grammaticale de chaque phrase. La production de cet arbre permet de reconnaître le mot `borrow` comme un verbe.

L'arbre syntaxique de la première phrase est donné à titre d'illustration dans l'annexe I.

5.5.2 Phase de reconnaissance

L'application des règles de détermination des rôles permet de caractériser sémantiquement chaque fait linguistique :

Pour la première phrase :
(`subscriber`, POSSESSEUR) ; (`have`, POS-SUJ) ; (`name`, PROPRIETE) ; (`address`, PROPRIETE)

Pour la seconde phrase :
(`book`, POSSESSEUR) ; (`identifie`, IDENT-SUJ); (`number`, IDENTIFIANT)

Pour la troisième phrase :
(`subscriber`, ACTEUR, CONTRAINT) ; (`borrow`, ACTION) ; (`at-most`, CONTRAINTE) ; (`book`, OBJET)

5.5.3 Phase d'interprétation

L'application des règles d'interprétation aboutit au schéma sémantique élémentaire présenté à la figure 5.17. Les cardinalités en caractère **gras** ont été obtenues à partir des règles de détermination des cardinalités tandis que les cardinalités en caractère maigre ont été obtenues à partir des règles interprétant les schémas (valeurs par défaut).

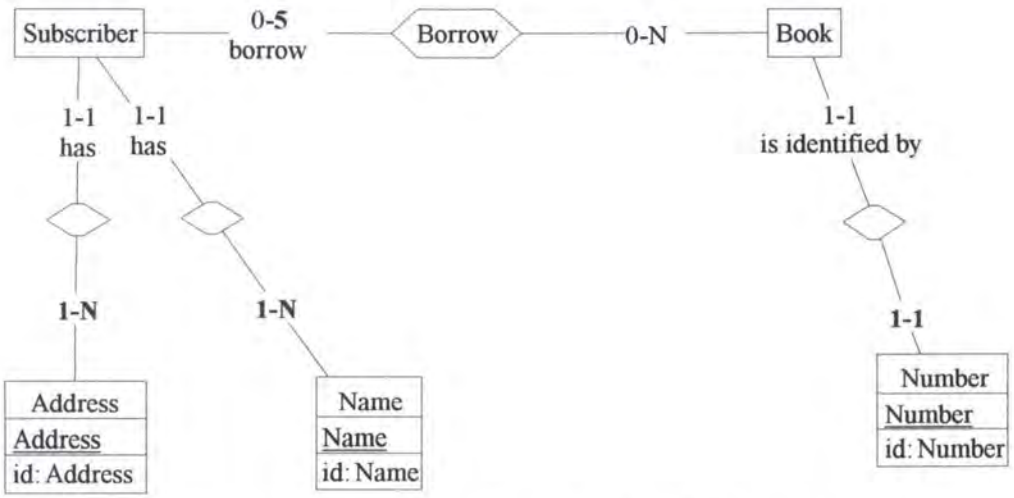


Figure 5.17. Schéma sémantique élémentaire obtenu par application des règles d'interprétation

6. Outils logiciels

6.1 Introduction

Dans ce chapitre, nous présentons les **outils logiciels** qui supportent notre démarche (cfr. figure 6.1). Ils ont pour vocation d'**assister** l'analyste durant tout le processus de conception d'un schéma conceptuel (cfr. chapitre 3).

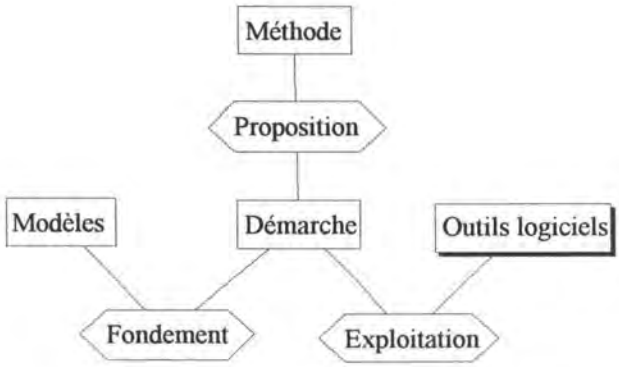


Figure 6.1. Méthode et outils logiciels

Nos outils logiciels sont une **aide active** à la construction de schémas conceptuels de données. Comme nous le verrons plus loin, cette aide intervient :

1. Durant la phase d'**élaboration du schéma sémantique élémentaire** ; c'est-à-dire durant la phase de représentation des éléments d'une phrase élémentaire en objets du modèle sémantique élémentaire.
2. Durant la phase de **transformation du schéma sémantique élémentaire en schéma EA de base**.
3. Durant la phase de **validation du schéma**.

Dans le paragraphe 6.2, nous présentons les outils logiciels qui supportent notre démarche. Les paragraphes suivants contiennent les spécifications des outils logiciels que nous avons développés.

6.2 Présentation générale des outils logiciels

Les outils logiciels que nous proposons permettent de passer progressivement des faits initiaux, exprimés sous la forme de phrases élémentaires, aux éléments du schéma EA de base. Ils mettent à la disposition de l'analyste deux environnements logiciels :

1. **un environnement en langage naturel** assurant d'une part, la saisie des descriptions initiales et d'autre part, le dialogue durant la phase de conception ;
2. **un environnement graphique** permettant d'interagir directement sur le contenu des schémas conceptuels présentés sous leur forme graphique.

L'**environnement en langage naturel** est proposé par l'outil logiciel **NATURAL EDITOR.** Il présente une interface en langage naturel assurant d'une part la **saisie du texte structuré** de phrases élémentaires et d'autre part, la **communication** des résultats obtenus durant le processus de conception (sous forme d'un rapport rédigé en langage naturel). Il supporte également la **phase de représentation** (cfr. chapitre 5, paragraphe 5.2).

L'**environnement graphique** est offert par l'atelier **DB-MAIN**. DB-MAIN ([HAINAUT, 96]) a été développé à l'Institut d'Informatique des Facultés Universitaires Notre-Dame de la Paix de Namur par l'équipe du Professeur J.-L. Hainaut. DB-MAIN est un outil CASE dédié à l'ingénierie de bases de données et plus particulièrement à la conception des bases de données, au *reverse engineering* et à la maintenance des bases de données. DB-MAIN possède de nombreuses fonctionnalités. Dans le cadre de notre méthode, nous n'utilisons qu'une partie de celles-ci :

1. conception et gestion de schémas sémantiques élémentaires et schémas EA de base ;
2. mécanismes de transformation d'un schéma sémantique élémentaire en schéma EA de base.

Intégrés dans DB-MAIN, les outils logiciels NATURAL DB-MAIN I et NATURAL DB-MAIN II proposent d'automatiser certaines phases du processus de conception. **NATURAL DB-MAIN I** réalise les phases de reconnaissance et d'interprétation (cfr. chapitre 5, paragraphes 5.3 et 5.4). **NATURAL DB-MAIN II** supporte l'analyste dans la phase de validation du schéma EA de base. Plus précisément, il vérifie si le schéma EA de base ne présente pas des structures à problème (TE sans attribut, TE sans identifiant, par exemple) et des cardinalités indéterminées (cfr. chapitre 3, paragraphe 3.5).

La figure 6.2 présente l'architecture globale des outils logiciels. Avant de présenter les outils logiciels que nous avons développés, nous présentons les différents fichiers manipulés et/ou échangés par ceux-ci.

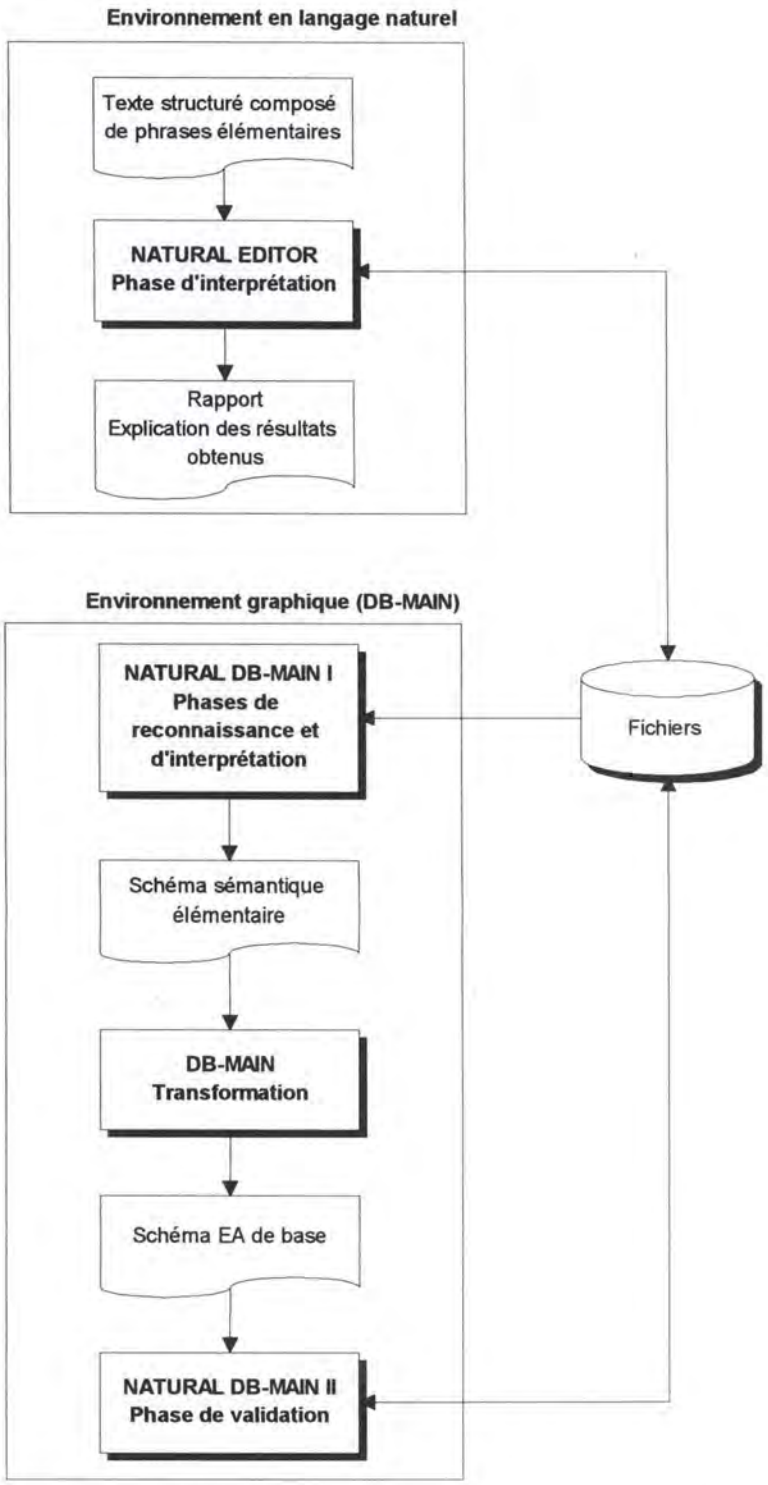


Figure 6.2. Présentation générale des outils logiciels

6.2.1 Fichiers manipulés par les outils logiciels

L'ensemble des éléments concourant à la conception d'un schéma conceptuel se contrôle à partir d'un **projet**. Un projet est identifié par un nom (`NomProjet`) rassemble un certain nombre de fichiers :

1. Le fichier **Source** (`NomProjet.TXT`). Ce fichier renferme le texte décrivant le domaine d'application.
2. Le fichier **Rapport** (`NomProjet.RAP`). Ce fichier conserve le contenu du rapport associé au projet. C'est dans ce fichier que NATURAL EDITOR stocke les résultats de la phase de représentation (erreur grammaticale, verbe absent du lexique, par exemple) et que NATURAL DB-MAIN II enregistre les structures à problème présentes dans le schéma EA de base (TE sans identifiant, par exemple).
3. Le fichier **Arbre**[3] (`NomProjet.OUT`). Ce fichier est un fichier de format texte dans lequel NATURAL EDITOR stocke l'arbre syntaxique, résultat de la phase de représentation. Ce fichier est utilisé par NATURAL DB-MAIN I pour construire le schéma sémantique élémentaire.

Pour connaître le nom du projet en cours, NATURAL DB-MAIN I & II disposent du fichier **Environnement**[4] (`NATURAL.INI`), présent dans le répertoire `C:\WINDOWS\`. Il contient le nom (`NomProjet`) et le chemin d'accès du projet en cours.

Pour supporter la phase de représentation, NATURAL EDITOR dispose du fichier **Lexique**[5]. Ce fichier contient la connaissance de notre outil logiciel sur le vocabulaire et la grammaire anglaise.

6.2.2 NATURAL EDITOR

NATURAL EDITOR supporte la **phase de représentation** (cfr. chapitre 5, paragraphe 5.2) et stocke la structure grammaticale des phrases (l'arbre syntaxique) dans le fichier **Arbre.**

NATURAL EDITOR présente une interface en langage naturel assurant d'une part la **saisie du texte structuré** de phrases élémentaires et d'autre part, la **communication** des résultats obtenus durant le processus de conception (sous forme d'un rapport rédigé en langage naturel). Il dispose également d'une interface **menu** permettant de sélectionner et d'activer les différentes fonctionnalités[6] proposées par l'outil logiciel. Il offre enfin une interface permettant d'interagir directement avec le lexique : consultation, ajout, retrait, modification de mots.

NATURAL EDITOR met à la disposition de l'analyste plusieurs fonctionnalités. Compte tenu du nombre élevé de ces fonctionnalités, nous nous limiterons à l'exposé succinct des plus

---

[3] La syntaxe du fichier **Arbre** est présentée dans l'annexe III.
[4] La syntaxe du fichier **Environnement** est présentée dans l'annexe IV.
[5] La syntaxe du fichier **Lexique** est présentée dans l'annexe V.
[6] Pour une description détaillée de toutes les fonctionnalités proposées par NATURAL EDITOR, nous renvoyons le lecteur à l'aide électronique.

usuelles. Notons toutefois que toutes les fonctionnalités sont décrites dans l'aide en ligne accompagnant l'outil logiciel.

Les principales fonctionnalités que propose NATURAL EDITOR sont les suivantes :

1. Ouverture d'un projet : prépare NATURAL EDITOR à recevoir le texte structuré.
2. Enregistrement d'un projet : sauvegarde le texte structuré dans le fichier **Source**, le rapport dans le fichier **Report** et met à jour le fichier **Environnement** (contenant le nom du projet en cours).
3. Récupération d'un ancien projet : lit les fichiers **Texte** et **Source** associés au projet à récupérer et affiche le contenu de ces fichiers dans l'interface en langage naturel.
4. Saisie du texte structuré.
5. Consultation du lexique : affiche le lexique et permet de rechercher un mot.
6. Ajout, modification, suppression d'un mot appartenant au lexique.
7. Consultation du rapport.
8. Support à la phase de représentation : contrôle la phase de représentation (cfr. chapitre 5, paragraphe 5.2).

Pour contrôler la phase de représentation, NATURAL EDITOR propose les fonctionnalités suivantes :

1. Contrôle de la grammaire du texte : procède aux analyses morphologiques, lexicales et syntaxiques.
2. Ajout des verbes absents du lexique : ajoute les verbes absents du lexique en précisant leur sens ainsi que leur forme primitive (substantif). L'ajout peut se faire sans ou avec l'intervention de l'analyste.
3. Communication des résultats obtenus : affiche, dans l'interface en langage naturel, les erreurs grammaticales et les verbes ajoutés.
4. Construction du fichier **Arbre**.

6.2.3 NATURAL DB-MAIN I

NATURAL DB-MAIN I couvre les phases de reconnaissance et d'interprétation décrites au chapitre 5. Il a pour objectif de construire un schéma sémantique élémentaire à partir de l'arbre syntaxique stocké dans le fichier **Arbre**. Pour connaître le nom du projet en cours, il consulte le fichier **Environnement**.

NATURAL DB-MAIN I travaille dans l'environnement DB-MAIN : il ouvre un nouveau projet et construit le schéma sémantique élémentaire par application des règles d'interprétation définies au chapitre 5, paragraphe 5.4.

Les principales actions effectuées sont communiquées à l'analyste : création du projet, création du schéma, création progressive du schéma sémantique élémentaire par insertions de types d'entité, de types d'association, etc.

6.2.4 DB-MAIN

La transformation du schéma sémantique élémentaire est exécutée automatiquement par l'atelier DB-MAIN qui propose un assistant de transformation globale, le **Global Transformations**. Cet assistant permet de transformer les TE propriétés en attribut et les TE entités en TA (cfr. chapitre 3, paragraphe 3.4). Le résultat de ces transformations est le schéma EA de base.

6.2.5 NATURAL DB-MAIN II

NATURAL DB-MAIN II supporte la phase de validation formelle : il applique les règles définies au chapitre 3, paragraphe 3.5.1 et produit un rapport contenant les structures à problèmes présentes dans le texte.

Il consulte, en entrée, le schéma EA de base pour en extraire les structures à problème (TE sans identifiant, TE sans attribut et TA similaires) et les cardinalités indéterminées. Il utilise également le fichier **Environnement** pour connaître le nom du projet en cours. Il enregistre enfin la liste des structures à problème et les cardinalités indéterminées dans le ficher **Report**. Ce fichier pourra être consulté par l'analyste via NATURAL EDITOR.

6.3 NATURAL EDITOR

Pour rappel, NATURAL EDITOR présente une interface en langage naturel assurant d'une part la **saisie du texte structuré** de phrases élémentaires et d'autre part, la **communication** des résultats obtenus durant le processus de conception (sous forme d'un rapport rédigé en langage naturel). Il supporte également la **phase de représentation** (cfr. chapitre 5, paragraphe 5.2) et stocke la structure grammaticale des phrases (l'arbre syntaxique) dans le fichier **Arbre**.

Ce chapitre présente l'architecture de NATURAL DB-MAIN et explique comment nous avons implanté la phase de représentation.

6.3.1 Architecture de NATURAL EDITOR

NATURAL EDITOR est composé de six modules (cfr. figure 6.3) :

1. Trois interfaces ;
2. le gestionnaire de projet ;
3. le module de représentation ;
4. le gestionnaire du lexique.

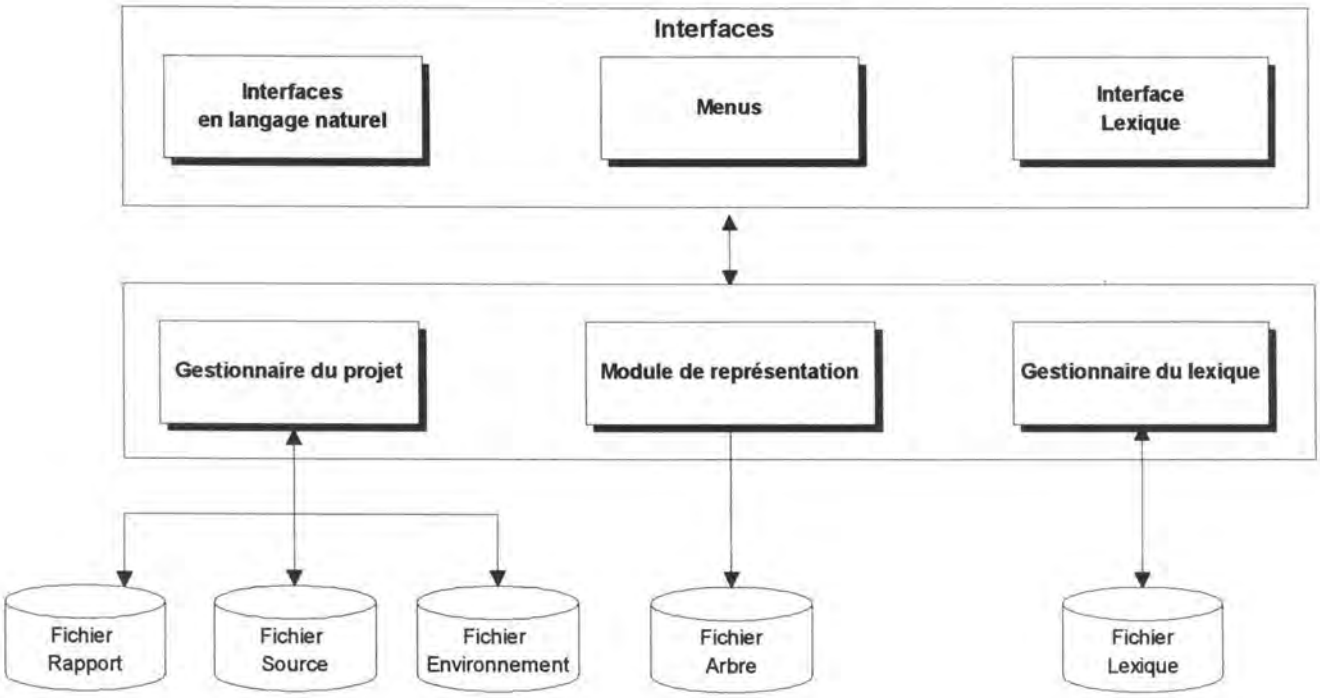


Figure 6.3. Architecture de NATURAL EDITOR

Le **interfaces** sont de trois types :

1. une interface **menu** permettant de sélectionner et d'activer les différentes fonctions de l'outil logiciel ;
2. une interface en **langage naturel** assurant d'une part, la saisie du texte structuré de phrases élémentaires et, d'autre part, la communication des résultats obtenus pendant la phase de conception ;
3. une interface **lexique** permettant d'interagir directement sur le lexique : consultation, ajout, retrait, modification de mots.

Le **gestionnaire du lexique** met à la disposition des autres modules et des interfaces les informations contenues dans le fichier **Lexique**[7]. Le gestionnaire du lexique offre quatre fonctionnalités :

1. **Déchargement du fichier Lexique** : il construit une représentation interne du fichier **Lexique**, plus commode et plus rapide pour les traitements ultérieurs (recherche d'un mot, modification des informations contenues dans le lexique, par exemple).
2. **Modification du lexique** : il met à jour la représentation interne du lexique lors d'un ajout, d'un retrait de mots ou encore lors d'une modification des informations relatives à un mot (modification du sens ou du substantif d'un verbe, modification de la nature grammaticale d'un mot, par exemple).
3. **Recherche d'un mot dans le lexique** : il parcourt le lexique pour obtenir les informations relatives à un mot cherché.

[7] La structure du fichier **Lexique** est présentée en détail dans l'annexe V.

4. **Chargement du fichier Lexique** : il enregistre la représentation interne du lexique dans le fichier **Lexique.**

Le **gestionnaire de projet** a pour rôle de gérer le projet. Pour réaliser cette tâche, il gère les fichiers **Environnement**, **Source** et **Rapport** et les variables globales liées au projet en cours. Il offre trois fonctionnalités :

1. **Ouverture d'un nouveau projet** : il reçoit le nom (variable globale `ProjectName`) et le chemin d'accès (variable globale `ProjectDir`) du nouveau projet. Il initialise la variable globale `Text` devant contenir le texte descriptif du domaine d'application et la variable globale `Report` devant contenir les résultats de la conception. Les variables globales `Text` et `Report` peuvent être considérées comme des chaînes de caractères.
2. **Enregistrement du projet en cours** : il enregistre le nom du projet (contenu `ProjectName`) et son chemin d'accès (contenu dans `ProjectDir`) dans le fichier Environnement ; il enregistre la variable `Text` dans le fichier **Source** et la variable `Report` dans le fichier **Rapport**.
3. **Récupération d'un ancien projet** : il reçoit le nom de l'ancien projet et son chemin d'accès. Il lit les fichiers **Source** et **Rapport** correspondants et place le contenu du fichier **Source** dans la variable `Text` et celui du fichier **Rapport** dans la variable `Report`.

Le module **d'interprétation** a pour rôle d'effectuer la phase de représentation. C'est le module principal de NATURAL EDITOR. Le paragraphe suivant présente les point importants de ce module.

6.3.2 Module de représentation

Le module de représentation a pour but de construire une représentation interne des phrases sous forme d'arbres syntaxiques mettant en évidence leur structure grammaticale. Ce module réalise les analyses morphologiques, lexicales et syntaxiques décrites au chapitre 5, paragraphe 5.2.

Le module de représentation utilise en entrée la variable globale `Text` contenant le texte à analyser et fournit, en sortie, le fichier **Arbre** contenant l'arbre syntaxique. Le module de représentation travaille phrase par phrase et est constitué de cinq composants principaux :

1. **Le séparateur de mots** : il reconnaît une phrase et isole les mots de cette phrase ;
2. **L'analyseur morphologique** : il met les mots sous leur forme canonique ;
3. **L'analyseur lexical** : il cherche les mots mis sous leur forme canonique dans le lexique et associe aux mots trouvés les informations contenues dans le lexique.
4. **L'analyseur syntaxique** : il vérifie d'une part que les phrases respectent bien la grammaire et d'autre part construit l'arbre syntaxique qui représente la structure grammaticale de la phrase ;
5. **Le transcripteur** : il construit une représentation de l'arbre syntaxique comme une suite de caractères et la stocke dans le fichier **Arbre**.

Toute incohérence détectée par le module de représentation est placée dans la variable globale `Report` et communiquée à l'analyste via l'interface en langage naturel. De plus, le module de représentation offre la possibilité :

1. soit, d'**automatiser** totalement la phase de représentation. Les mots reconnus comme verbe par l'analyseur syntaxique sont ajoutés dans le lexique en tant que verbe d'action (cfr. chapitre 5, paragraphe 5.2) ;
2. soit de déclencher un **dialogue interactif** lorsqu'un mot reconnu comme verbe par l'analyseur syntaxique est absent du lexique. Ce dialogue permet à l'analyste d'affecter lui même le sens du verbe.

D'autre part, le module de représentation peut réagir de deux manières différentes lorsque l'analyseur syntaxique rencontre une erreur syntaxique :

1. soit il arrête la phase de représentation ;
2. soit il ignore l'erreur et poursuit la phase de représentation à la phrase suivante.

Toute erreur grammaticale détectée par le module est placée dans la variable globale `Report` et provoque un message d'erreur. L'analyste est alors invité à corriger les éventuelles erreurs grammaticales et à relancer le module de représentation.

Au sein du module de représentation, les mots d'une phrase sont représentés sous la forme d'une liste renfermant les informations nécessaires à la construction du fichier **Arbre**. Avant de spécifier chaque composant du module de représentation, nous décrivons comment se présente une phrase vue comme une liste de mots.

6.3.2.1 Représentation de la phrase sous forme d'une liste de mots

Chaque composant du module de représentation participe à la représentation de la phrase sous forme d'une liste renfermant les informations nécessaires à la construction du fichier **Arbre.**

La phrase est représentée par une liste (`ListeMot`) qui se présente comme suit :

```
[Mot_1, Mot_2, ..., Mot_n]
```

Un `Mot_i` est une liste contenant les informations associées à un mot de la phrase. `Mot_i` se présente comme suit :

```
[MotRéel, MotCanonique, NatureGram, Information, Sens,
 NumeroGN, GroupeGram]
```

où

- `MotRéel` est une chaîne de caractères représentant le mot tel qu'il apparaît dans la phrase. Ce champ est complété par le séparateur de mot.
- `MotCanonique` est une chaîne de caractères représentant le mot sous sa forme canonique. Ce champ est complété par l'analyseur morphologique.
- `NatureGram` est une chaîne de caractères représentant la nature grammaticale du mot. Ce champ est complété par l'analyseur lexical ;

- `Information` est une chaîne de caractères représentant le substantif du verbe. Ce champ est complété par l'analyseur lexical ;
- `Sens` est une chaîne de caractères représentant la classe sémantique si c'est un verbe (cfr. chapitre 5, paragraphe 5.3.1), la catégorie du mot si c'est un mot-clé (cfr. chapitre 5, paragraphe 5.2.1). Ce champ peut être complété par l'analyseur lexical ou syntaxique.
- `GroupeGram` est une chaîne de caractères représentant le groupe grammatical (groupe sujet, groupe verbal, groupe complément ou groupe informatif) auquel appartient le mot. Ce champ est complété par l'analyseur syntaxique.
- `NumeroGN` est un entier représentant le numéro du groupe nominal dans son groupe grammatical. Ce champ est complété par le transcripteur.

6.3.2.2 Séparateur de mots

Le séparateur de mots est constitué de la seule procédure `LireMot`. Elle a pour objectifs :

1. de lire et reconnaître une phrase du texte ;
2. d'isoler les mots de cette phrase ;
3. de stocker le numéro et la forme fléchie de chaque mot dans la variable `ListeMot`.

Avant de spécifier l'analyseur morphologique, il est nécessaire de définir comment se présente une phrase vue comme une chaîne de caractères terminée par un point.

6.3.2.2.1 Représentation d'une phrase sous forme d'une chaîne de caractères

Toute phrase est définie, d'un point de vue morphologique, comme étant une suite de mots séparés par un espace et terminée par un point. La spécification d'une phrase est précisée ci-dessous sous forme BNF[8] :

```
Phrase     ::= Mot {Espace Mot} FinPhrase
Mot        ::= ChaineCar
FinPhrase  ::= .
Espace     ::= <espace>|,|;
```

Le concept de `ChaineCar` est défini ci-dessous comme étant une suite de **caractères de base** ou une suite de **chiffres** :

```
ChaineCar  ::= CarBase{CarBase}|Chiffre{Chiffre}
CarBase    ::= Lettre|CarSpecial
Lettre     ::= LettreMaj|LettreMin
LettreMaj  ::= A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z
LettreMin  ::= a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z
CarSpecial ::= -|'
Chiffre    ::= 1|2|3|4|5|6|7|8|9|0
```

[8] Les conventions BNF utilisées sont présentées dans l'annexe II.

6.3.2.2.2 Spécification du séparateur de mots

Pour rappel, le séparateur de mots est constitué de la seule procédure `LireMot`. Elle a pour objectif de lire une phrase contenue dans la variable `Text` et d'isoler les mots de cette phrase. Le séparateur de mot lit, caractère par caractère, la variable `Text`.

Voici la spécification de `LireMot` :

Nous désignons par **caractère valide** un caractère qui est un caractère de base ou un chiffre. Nous appelons **position courante** de lecture de la variable `Text` la position du prochain caractère à lire par le séparateur de mots.

Soit `C` la suite de caractères restant à lire dans la variable `Text` (la position courante est le premier caractère de `C`). `C` se décompose en `C'D` où `C'` est une suite, peut être vide, de caractères et où `D` contient un point ou une suite, peut être vide, d'espaces.

> Si `C'` et `D` sont vides (on a atteint la fin du texte), alors la phase de représentation est terminée.
> Si `C'` est une suite non vide de caractères valides et `D` un espace, alors `C'` est un mot et `LireMot` ajoute `C'` dans `ListeMots`. Il place la position de lecture sur le caractère suivant `D` et continue à lire la variable `Text`.
> Si `C'` est une suite non vide de caractères valides et `D` un point (on a atteint la fin de la phrase), alors `C'` est le dernier mot de la phrase. `LireMot` ajoute `C'` dans `ListeMots` et place la position courante de lecture sur le caractère suivant `D` et arrête la lecture de la variable `Text`.

Toute incohérence détectée par le séparateur de mots (`C'` contient des caractères non valides) est placée dans la variable globale `Report` et provoque l'arrêt de la phase de représentation. L'analyste est alors invité à corriger les éventuelles erreurs et à relancer la phase de représentation.

6.3.2.3 Analyseur morphologique

L'analyseur morphologique a pour objectif d'isoler la forme canonique des mots de la phrase courante. Il est constitué de la seule fonction `Canonique` qui reçoit, en paramètre, un mot de `ListeMots` et renvoie sa forme canonique.

Pour isoler la forme canonique d'un mot, la fonction `Canonique` met en œuvre un mécanisme d'analyse et de génération pour reconnaître la forme stockée d'un mot (c'est-à-dire sa forme canonique) à partir de sa forme fléchie. Elle est basée sur **l'analyseur procédural** de Winograd ([SABAH, 90a]). L'analyseur procédural procède à une analyse morphologique basée sur des règles correspondantes à des actions à effectuer pour retrouver la forme canonique à partir d'une forme fléchie. Ces règles consistent à supprimer certaines fins de mots (comme `ing`, `ed` ou `en` pour les verbes, `'s` ou `s` pour les noms) et ajouter certaines lettres, pour reconstruire la forme connue (ajouter `an` après avoir ôté `en` de `men`, pour analyser le pluriel, par exemple).

L'analyseur procédural ne couvre pas la totalité de la langue anglaise, mais une bonne partie de celle-ci et selon Winograd, il peut être facilement étendu pour tenir compte des cas particuliers qui n'entrent pas dans l'algorithme général.

6.3.2.4 Analyseur lexical

L'analyseur lexical a pour objectif de chercher les mots de `ListeMots` dans le lexique et de placer les informations relatives à ces mots dans `ListeMots`. L'analyseur lexical procède mot par mot et utilise la procédure `RECHERCHE_MOT` du gestionnaire du lexique. Deux cas peuvent se présenter :

1. Le mot est présent dans le lexique. Dans ce cas, `RECHERCHE_MOT` renvoie les informations stockées dans le lexique pour ce mot (la nature grammaticale, l'information associée et le sens) et l'analyseur lexical place ces informations dans `ListeMots`.
2. Le mot est absent dans le lexique. Dans ce cas, `RECHERCHE_MOT` ne renvoie aucune donnée. C'est à l'analyseur syntaxique de déterminer la nature grammaticale de ce mot.

6.3.2.5 Analyseur syntaxique

L'analyseur syntaxique a pour objectif :

1. de vérifier que la phrase soit grammaticalement correcte ;
2. de déterminer le groupe grammatical à laquelle appartiennent les mots (groupe sujet, groupe verbal, groupe complément ou groupe informatif) afin de compléter la variable `ListeMots`;
3. d'affecter une valeur sémantique aux verbes absents du lexique.

L'analyseur syntaxique travaille sur les mots contenus dans `ListeMots` et applique les règles syntaxiques définies au chapitre 5, paragraphe 5.2.2). La stratégie que nous employons consiste à :

1. Décomposer la phrase en un groupe sujet, groupe verbal, groupe informatif et groupe complément. Le groupe sujet est assimilé au premier groupe de mots de `ListeMots`.
2. Appliquer les règles de production aux différents groupes de la phrase en commençant par le groupe le plus à gauche. Ainsi, par exemple, la nature grammaticale du premier mot d'une phrase peut être un article ou un nom.
3. Si un mot est reconnu comme `<verbe inconnu>`, l'ajouter dans le lexique en tant que verbe d'action ou déclencher le dialogue interactif.
4. Une fois que le groupe grammatical auquel appartient un mot est reconnu, le placer dans `ListeMots`.

Toute erreur grammaticale détectée est placée dans la variable globale `Report` et provoque l'arrêt de l'analyse syntaxique ou la poursuite de l'analyse à la phrase suivante. L'analyste est alors invité à corriger les éventuelles erreurs grammaticales et à relancer la phase de représentation.

6.3.2.6 Transcripteur

Le transcripteur est composé d'une seule procédure `TRANSCRIRE` qui a pour objectif de lire la variable `ListeMots` et de construire une **représentation externe** de l'arbre syntaxique de la phrase. Celle-ci est ensuite stockée dans le fichier **Arbre**. La représentation externe de l'arbre syntaxique est une chaîne de caractères représentant l'arbre syntaxique de la phrase et respectant la syntaxe propre au fichier **Arbre** et définie dans l'annexe III.

6.4 NATURAL DB-MAIN I

NATURAL DB-MAIN I couvre les phases de reconnaissance et d'interprétation décrites au chapitre 5. Il a pour objectif de construire un schéma sémantique élémentaire à partir d'un arbre syntaxique stocké dans le fichier **Arbre**.

NATURAL DB-MAIN a été implanté dans l'atelier DB-MAIN et rédigé entièrement en VOYAGER II. VOYAGER II est un langage de programmation qui accompagne DB-MAIN et est comparable aux langages classiques tel que C ou PASCAL ([ENGLEBERT, 97]). Il offre des fonctions qui permettent de construire directement un schéma sémantique élémentaire : création de TE, d'attributs, etc.

6.4.1 Architecture de NATURAL DB-MAIN I

Il est constitué de quatre modules principaux :

1. **le module de préparation** : prépare l'atelier DB-MAIN à l'exécution des autres modules ;
2. **le module de transfert** : construit une représentation interne du fichier **Arbre** ;
3. **le module de reconnaissance** : caractérise sémantiquement les arbres syntaxiques (cfr. chapitre 5, paragraphe 5.3) ;
4. **le module d'interprétation** : génère un schéma sémantique élémentaire (cfr. chapitre 5, paragraphe 5.4).

L'architecture globale de NATURAL DB-MAIN I est illustrée figure 6.4.

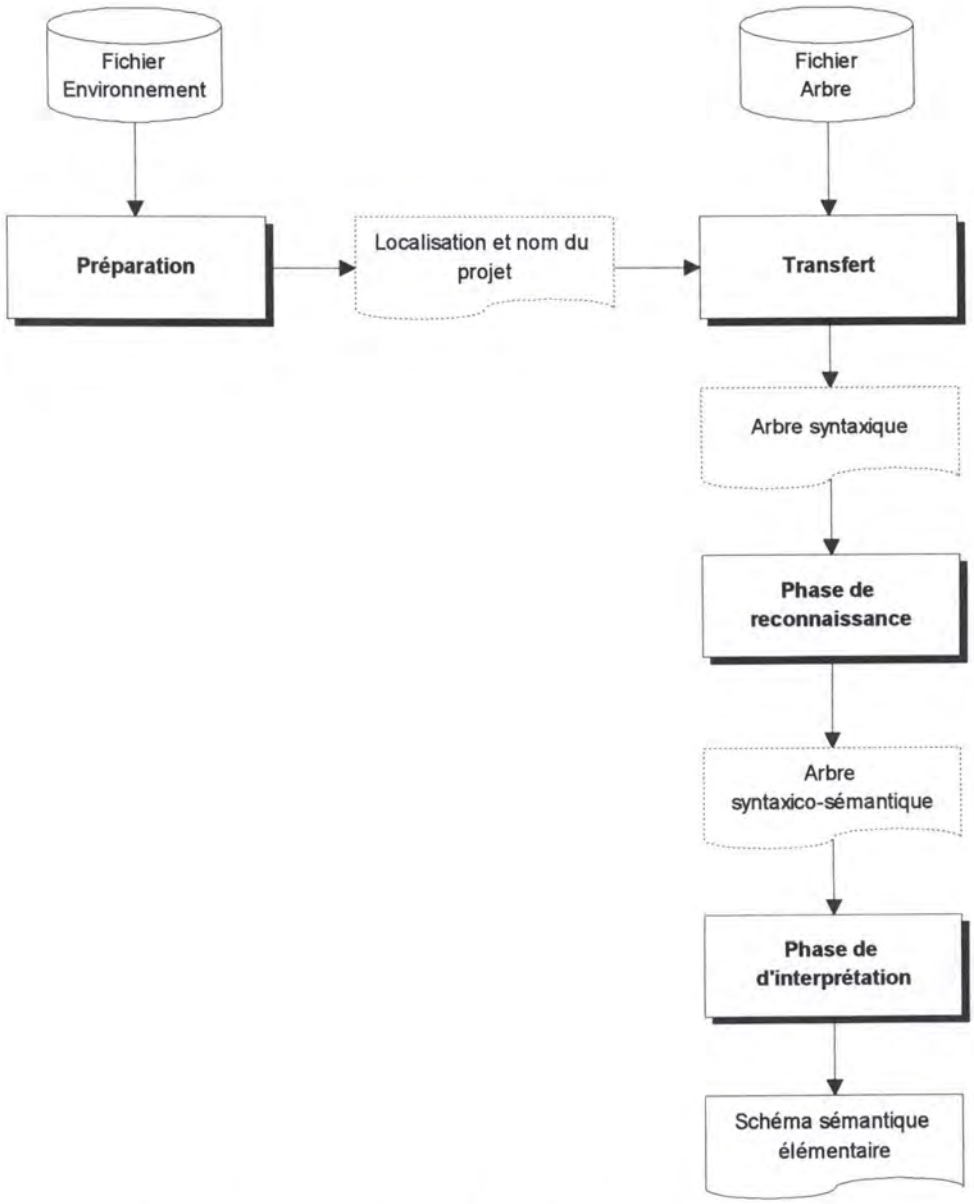


Figure 6.4. Architecture de NATURAL DB-MAIN I

Détaillons à présent chaque module en décrivant ses principales caractéristiques, les informations qu'il utilise et celles qu'il produit en sortie.

6.4.2 Module de préparation

Le module de préparation a pour objectif de préparer l'atelier DB-MAIN à l'exécution de NATURAL DB-MAIN I. Cette préparation consiste à :

1. déterminer le projet en cours ;
2. ouvrir un projet dans l'atelier DB-MAIN ;
3. ouvrir un schéma qui contiendra le schéma sémantique élémentaire ;
4. attacher aux cardinalités une propriété d'indétermination.

Avant de spécifier le module, nous définissons ce que nous entendons par propriété d'indétermination d'une cardinalité.

6.4.2.1 Propriété d'indétermination d'une cardinalité

Lors de la phase d'interprétation, notre outil logiciel utilise les règles standards et les règles de base pour déterminer la valeur des cardinalités (cfr. chapitre 5, paragraphe 5.4.2). Il existe cependant des cas où ces règles sont incapables de déterminer la valeur des cardinalités. Les règles spécifiques aux schémas structuraux et associatifs permettent alors d'attribuer aux cardinalités **indéterminées** des valeurs par défaut.

Pour rendre compte de cette indétermination, nous avons associé à chaque rôle deux **propriétés dynamiques**[9] : la propriété `MINIMUM` liée à la cardinalité minimale, et la propriété `MAXIMUM` liée à la cardinalité maximale. Ces propriétés peuvent prendre deux valeurs :

- `0` en cas d'indétermination (c'est-à-dire en cas de valeur par défaut) ;
- `1` en cas de cardinalité déterminée à partir des règles standards ou de base.

6.4.2.2 Spécification du module de préparation

Le module de spécification consulte en entrée le fichier **Environnement** pour extraire le nom et le chemin d'accès du projet en cours. Il ouvre ensuite un nouveau projet dans l'atelier DB-MAIN et un schéma qui contiendra le schéma sémantique élémentaire. Enfin, il ajoute, aux propriétés associées à un rôle, les propriétés dynamiques : `NAT_MINIMUM` et `NAT_MAXIMUM`. Ces valeurs sont initialisées à `0`.

6.4.3 Module de transfert

Le module de transfert est constitué d'une seule procédure `OBTENIR_ARBRE_PHRASE` qui a pour objectif de lire le fichier **Arbre** et de créer une **représentation interne** de l'arbre syntaxique. Toute incohérence syntaxique détectée (le fichier **Arbre** ne respecte pas la syntaxe définie dans l'annexe VI) par le module de transfert provoque l'arrêt de NATURAL DB-MAIN I. La représentation interne de l'arbre syntaxique est définie dans l'annexe VI.

6.4.4 Module de reconnaissance

Le module de reconnaissance a pour objectif de caractériser sémantiquement les arbres syntaxiques. Il détermine le sens réel du verbe, ainsi que le rôle de chaque mot. Le résultat de

[9] Les propriétés dynamiques permettent d'ajouter des nouvelles propriétés à un objet (type d'entité, attribut, rôle, etc.). Pour une définition précise et complète des propriétés dynamiques, nous renvoyons le lecteur à [ENGLEBERT, 97].

ce module est un arbre syntaxico-sémantique exprimant la sémantique et la syntaxe des phrases. La représentation de l'arbre syntaxico-sémantique est définie dans l'annexe VII.

Le module de reconnaissance procède en deux phases :

1. Dans une première phase (procédure `REGLE_ROLE_1`), il détermine **le sens réel des verbes**. Pour ce faire, il applique la `REGLE-ROLE-1` basée sur la présence ou l'absence d'auxiliaire être (cfr. chapitre 5 , paragraphe 5.3.1).
2. Dans une seconde phase (procédure `REGLE_ROLE_2_ET_PLUS`), il détermine les **rôles** de chaque élément de la phrase. La détermination des rôles est obtenue par application des huit dernières règles de détermination des rôles (cfr. chapitre 5, paragraphe 5.3.2).

6.4.5 Module d'interprétation

Le module d'interprétation a pour objectif de construire le schéma sémantique élémentaire à partir des informations contenues dans l'arbre syntaxico-sémantique (cfr. chapitre 5, paragraphe 5.4). La construction du schéma sémantique élémentaire est effectuée à partir des règles d'interprétation des schémas, des règles de détermination des cardinalités des T.A. et des règles déterminant les propriétés du type générique.

Avant de spécifier le module d'interprétation, nous présentons comment nous avons implanté les règles de détermination des cardinalités dans l'atelier DB-MAIN.

6.4.5.1 Détermination des cardinalités

Dans le chapitre 5, paragraphe 5.4.1, nous avons distingué deux classes de règles de détermination de cardinalités :

1. les **règles élémentaires**, les **règles de base** déterminant la valeur des cardinalités attachées au rôle CONTRAINT, IDENTIFIANT ou PROPRIETE ;
2. les **règles spécifiques** aux schémas attribuant une valeur **par défaut** aux cardinalités indéterminées par les premières règles.

Pour chaque cardinalité déterminée par la première classe de règles, le module d'interprétation attribue la valeur 1 à la propriété dynamique d'indétermination associée. Les règles spécifiques ne peuvent donc pas modifier la valeur d'une cardinalité associée à une propriété dynamique de valeur 1.

6.4.5.2 Spécification du module d'interprétation

Le modèle d'interprétation travaille sur l'arbre syntaxico-sémantique et procède phrase par phrase. Pour chaque phrase :

1. le module d'interprétation applique les règles d'interprétation des schémas (cfr. chapitre 5, paragraphe 5.4.1) ;

2. si la phrase s'unifie à un schéma associatif de spécialisation, il applique les règles de détermination des propriétés du type générique (cfr. chapitre 5, paragraphe 5.4.3) ;
3. si la phrase s'unifie à un autre type de schéma, il utilise les règles de détermination des cardinalités (cfr. chapitre 5, paragraphe 5.4.2) et met à jour les valeurs des propriétés dynamiques d'indétermination (cfr. chapitre 5, paragraphe 6.5.5.1).

6.5 NATURAL DB-MAIN II

NATURAL DB-MAIN II a pour objectif de vérifier si le schéma EA de base ne présente pas de structures à problème (TE sans identifiant, TE sans attribut, TA similaires) ou des cardinalités indéterminées (cfr. chapitre 3, paragraphe 3.5).

NATURAL DB-MAIN II a été implanté dans l'atelier DB-MAIN et rédigé en VOYAGER II. Grâce aux nombreuses fonctionnalités offertes par VOYAGER II, NATURAL DB-MAIN II peut facilement repérer certaines structures à problème et les cardinalités indéterminées (via les propriétés dynamiques attachées aux rôles).

6.5.1 Architecture de NATURAL DB-MAIN II

NATURAL DB-MAIN II est composé de deux modules :

1. **le module de préparation** : détermine le projet en cours. Il consulte le fichier **Environnement** pour extraire le nom et le chemin d'accès du projet en cours.
2. **le module de validation** : vérifie si le schéma EA de base ne contient pas des structures à problème ou des cardinalités indéterminées et enregistre ces éléments dans le fichier **Report.**

L'architecture est illustrée par la figure 6.5. Nous spécifions dans le paragraphe suivant le module de validation.

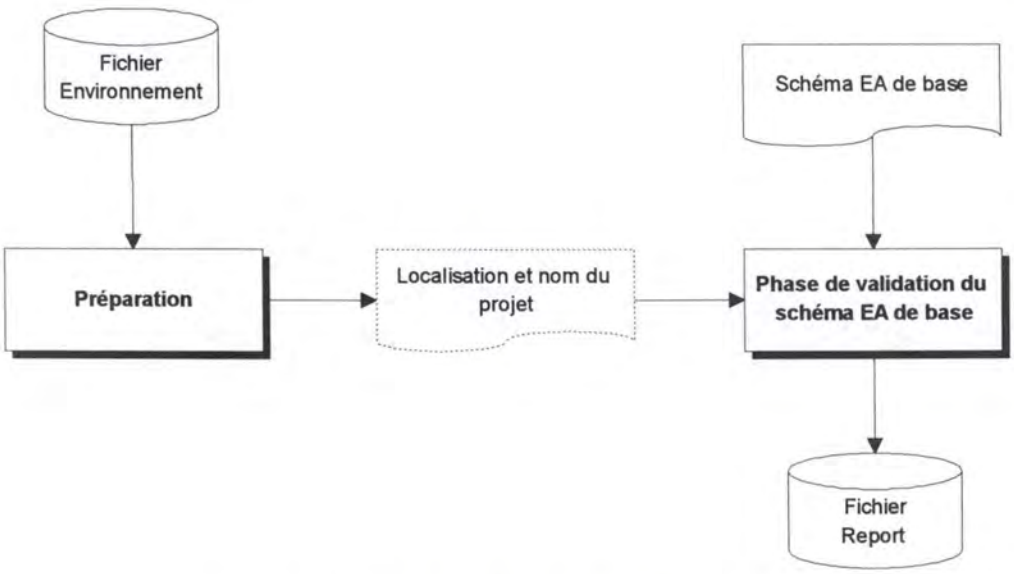


Figure 6.5. Architecture de NATURAL II

6.5.2 Module de validation

Le module de validation consulte en entrée le schéma EA de base pour constituer :

1. une liste contenant les TE sans attribut ;
2. une liste contenant les TE non identifiés ;
3. une liste contenant les rôles associés à au moins une cardinalité indéterminée, c'est-à-dire les rôles ayant une de leurs propriétés `NAT_MINIMUM` ou `NAT_MAXIMUM` à `0`;

A partir de ces listes, le module de validation utilise des structures de messages textes prédéfinies de manière à former des textes en langage naturel qui sont enregistrés dans le fichier **Report**.

7. Etude de cas

7.1 Introduction

Dans ce chapitre, nous proposons d'**appliquer** notre méthode à une étude de cas (cfr. figure 7.1). Cette étude porte sur la construction d'un schéma Entité-Association de base à partir d'un document écrit en anglais. Nous présentons les différentes **étapes** de notre **démarche** et les **outils logiciels** les supportant.

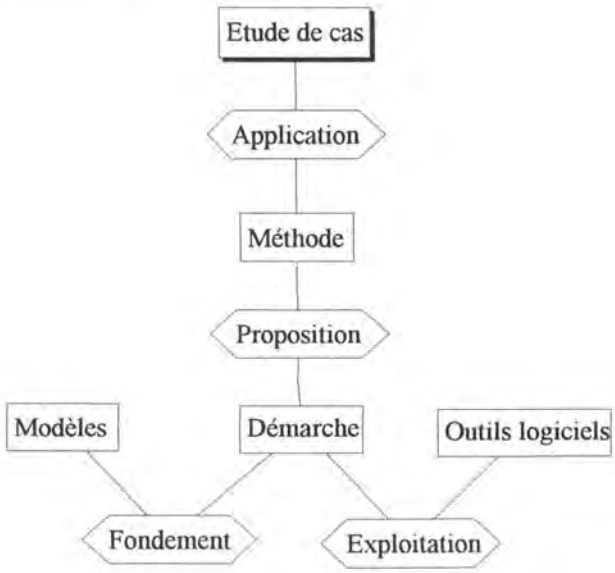


Figure 7.1. Méthode et Etude de cas

7.2 Description du cas

Le cas que nous proposons est basé sur une étude de gestion de bibliothèque. Il est une adaptation du cas d'école proposé dans [DB-MAIN, 95]. Le document qui suit est le résultat d'un entretien avec les employés de la bibliothèque.

A book is considered as a piece of literary, a scientific or technical document. It's identified by a number. Each book is characterized by its title, the first published date, keywords (maximum 6), an abstract and its bibliographic references. A book is also characterized by its physical state (new, used, worn, torn, damaged, etc). A technical document is characterized by a mandatory comment.

A book can be written by several authors. An author can have a first name, a birth date, and an origin (i.e., the organization

(s)he belongs to when the book was written). For some authors, only the name is known. To be recorded in the data base, an author must write at least one book.

For each book, the library has acquired a certain number (0, 1, or more) of copies. The copies have distinct serial numbers. For each copy, the date it was acquired is known as well as its location in the library.

A copy can be borrowed by only one borrower. Borrowers are identified by a personnal id. They are characterized by their name, their address (name of the company, street, zip-code and city name), and their phone numbers (at most five). A borrower can borrow at most five books.

7.3 Première étape : analyse de l'énoncé

Cette étape est consacrée à l'organisation de l'énoncé sous la forme d'un **texte structuré** en phrases élémentaires. Elle est entièrement prise en charge par l'**analyste** (cfr. figure 7.2). Il est essentiel de souligner l'importance de cette étape. En effet, l'essentiel du travail de **modélisation** est réalisé au cours de cette étape. La suite consiste simplement, pour l'analyste, à utiliser les outils logiciels que nous avons développés.

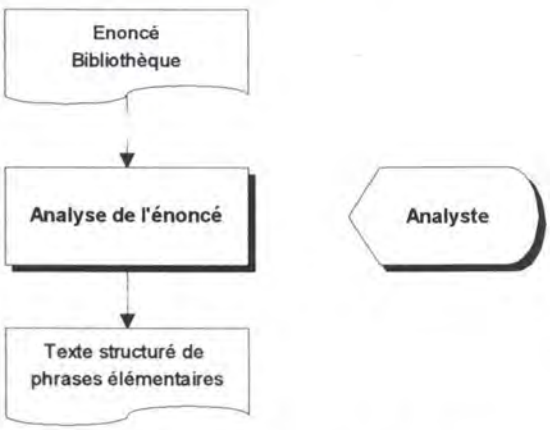


Figure 7.2. Analyse de l'énoncé prise en charge par l'analyste

Par souci de clarté, nous avons structuré le texte en quatre paragraphes, chacun décrivant un concept majeur du domaine d'application.

7.3.1 Books

(1) A book is considered as a piece of literary, a scientific or technical document.

La phrase est élémentaire mais ne respecte pas la grammaire imposée. Sans perte de sémantique, elle peut être remplacée par la phrase suivante :

```
A book can be a literary-document, a scientific-document
or a technical-document.
```

(2) `It's identified by a number.`

La phrase n'est pas élémentaire car elle contient un raccourci (le pronom `It`). Nous remplaçons le pronom par le type d'objet qu'il représente :

```
A book is identified by a number.
```

(3) `Each book is characterized by its title, the first published date, keywords (maximum 6), an abstract and its bibliographic references.`

La phrase n'est pas élémentaire et ne respecte pas la grammaire. Elle peut être décomposée comme suit :

```
Each book is characterized by its title, the first-
published-date, keywords, an abstract and
its bibliographic-references.
A book can have at-most 6 keywords.
```

(4) `A book is also characterized by its physical state (new, used, worn, torn, damaged, etc).`

La phrase n'est pas élémentaire. Elle peut être décomposée en constructions plus courtes, sans perte de sens :

```
A book is characterized by its physical-state.
```

(5) `A technical-document is characterized by a mandatory comment.`

La phrase ne respecte pas la grammaire (les adjectifs ne sont pas autorisés). Elle peut être remplacée par la phrase :

```
A technical-document must have a comment.
```

7.3.2 Authors

(6) `A book can be written by several authors.`

La phrase est correcte.

(7) `An author can have a first name, a birth date, and an origin (i.e., the organization(s)he belongs to from when the book was written).`

La phrase contient de l'information superflue : `the organization (s)he belongs to from when the book was written.` Exempte de cette information, la phrase devient :

```
An author can have a first-name, a birth-date, and an
origin.
```

(8) `For some authors, only the name is known.`

La phrase peut être simplifiée comme suit :

```
Each author has a name.
```

(9) `To be recorded in the data base, an author must write at`

least one book.
La phrase ne respecte pas la grammaire et contient de l'information superflue. Elle peut être remplacée, sans perte de sens, par la phrase suivante :
An author must write at-least 1 book.

7.3.3 Copies

(10) For each book, the library has acquired a certain number (0, 1, or more) of copies.
La phrase peut être exprimée par une construction plus explicite et plus simple :
A book can be represented by several copies.

(11) The copies have distinct serial numbers.
La phrase ne respecte pas la grammaire mais peut être remplacée par la phrase suivante :
The copies are identified by their ser-number.

(12) For each copy, the date it was acquired is known as well as its location in the library.
La phrase n'est pas élémentaire : elle peut être exprimée par les phrases suivantes :
Each copy is characterized by its date and its location.

7.3.4 Borrowers

(13) A copy can be borrowed by only one borrower.
(14) Borrowers are identified by a personnal-id.
Les phrase (13) et (14) sont correctes.

(15) They are characterized by their name, their address (at most five).
La phrase contient un pronom (construction implicite) et peut être décomposée en constructions plus courtes sans perte de sens :
Borrowers are characterized by their name.
Borrowers can have at-most 5 phone-numbers.

(16) A borrower can borrow at-most 5 books.
La phrase est correcte.

7.4 Deuxième étape : élaboration d'un schéma sémantique élémentaire

La production du schéma sémantique élémentaire se fait par transformation du texte structuré issu de l'étape précédente. Cette phase est entièrement supportée par les **outils logiciels** NATURAL EDITOR et NATURAL DB-MAIN I (cfr. figure 7.3).

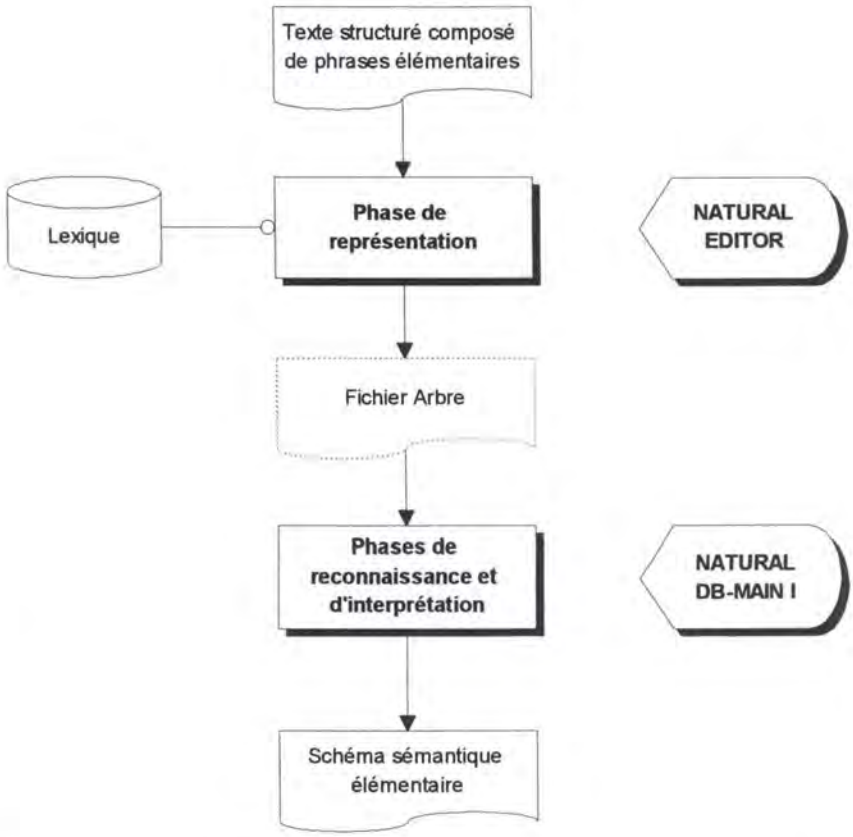


Figure 7.3. Elaboration d'un schéma sémantique élémentaire supportée par les outils logiciels NATURAL EDITOR et NATURAL DB-MAIN I

NATURAL EDITOR supporte la phase de représentation (cfr. chapitre 6, paragraphe 6.4) et stocke la structure grammaticale des phrases dans le fichier **Arbre**. NATURAL DB-MAIN I couvre les phases de reconnaissance et d'interprétation (cfr. chapitre 6, paragraphe 6.5) et génère automatiquement le **schéma sémantique élémentaire**.

7.4.1 Phase de représentation

NATURAL EDITOR présente une **interface en langage naturel** assurant :

1. la saisie du texte structuré ;
2. la compilation du texte, c'est-à-dire la reconnaissance et la détermination de la nature grammaticale de chaque mot ;
3. l'affectation du sens des verbes.

Les deux dernières opérations sont effectuées par référence à un lexique[10]. Le lexique représente la connaissance de notre outil logiciel sur le vocabulaire et la grammaire anglaise (cfr. chapitre 5, paragraphe 5.2). Les verbes d'**action** ne sont généralement pas présents car ils correspondent au vocabulaire particulier du domaine d'application. C'est à l'analyste de les inclure dans le lexique.

7.4.1.1 Présentation et préparation de NATURAL EDITOR

Après avoir lancé NATURAL EDITOR (`NATURAL.EXE`), la première tâche à faire est de créer un nouveau projet. Les commandes à exécuter sont les suivantes :

1. Dérouler le menu **Project** et sélectionner **New Project**. La boîte de dialogue **Create a New Project** (cfr. figure 7.4) est affichée.
2. Dans la zone **Name**, entrer le nom du projet : `LIBRARY`.
3. Cliquer sur le bouton **OK** pour valider l'opération.

[10] Le lexique que nous nous proposons d'utiliser dans cette étude de cas est décrit dans l'annexe VIII.

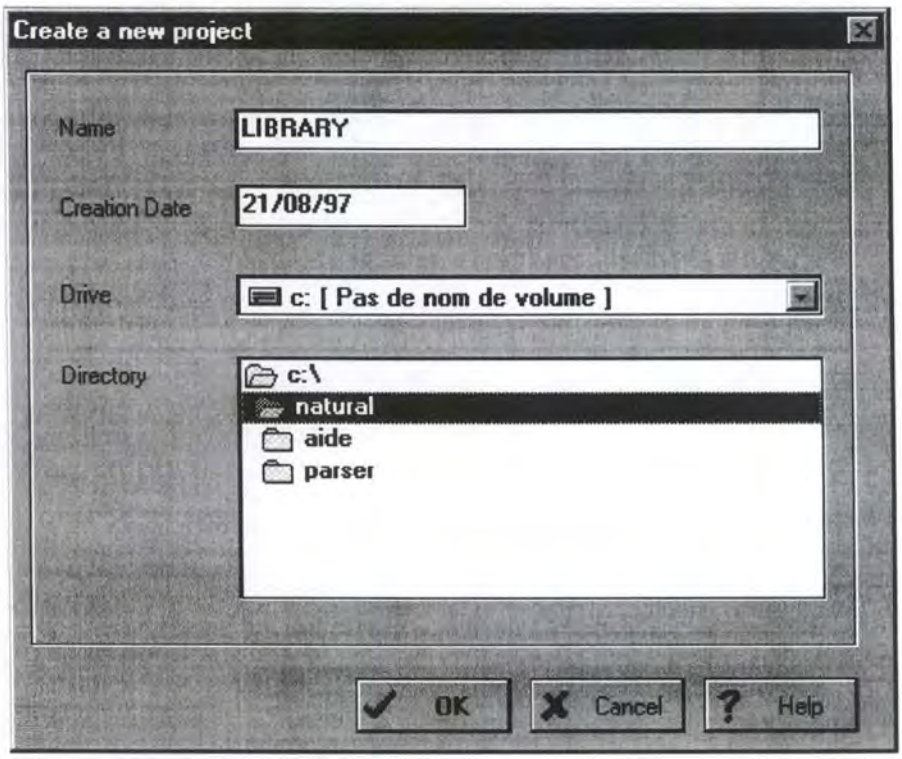


Figure 7.4. Boîte de dialogue Create a New Project

L'interface de NATURAL EDITOR, telle qu'elle apparaît après la création du projet, est présentée figure 7.5. Nous trouvons d'abord la **barre des menus**, en haut du bureau, sous laquelle se situe la **barre d'icônes**[11]. Au centre du bureau apparaissent deux fiches vierges appartenant au **Text Edit** et au **Report**. La fiche **Text Edit** est destinée à recevoir le texte décrivant le domaine d'application tandis que la fiche **Report** informera l'analyste sur les résultats de la compilation (erreur grammaticale, verbe absent du lexique, par exemple).

[11] Pour une description détaillée de toutes les fonctionnalités proposées par NATURAL EDITOR, nous renvoyons le lecteur à l'aide électronique.

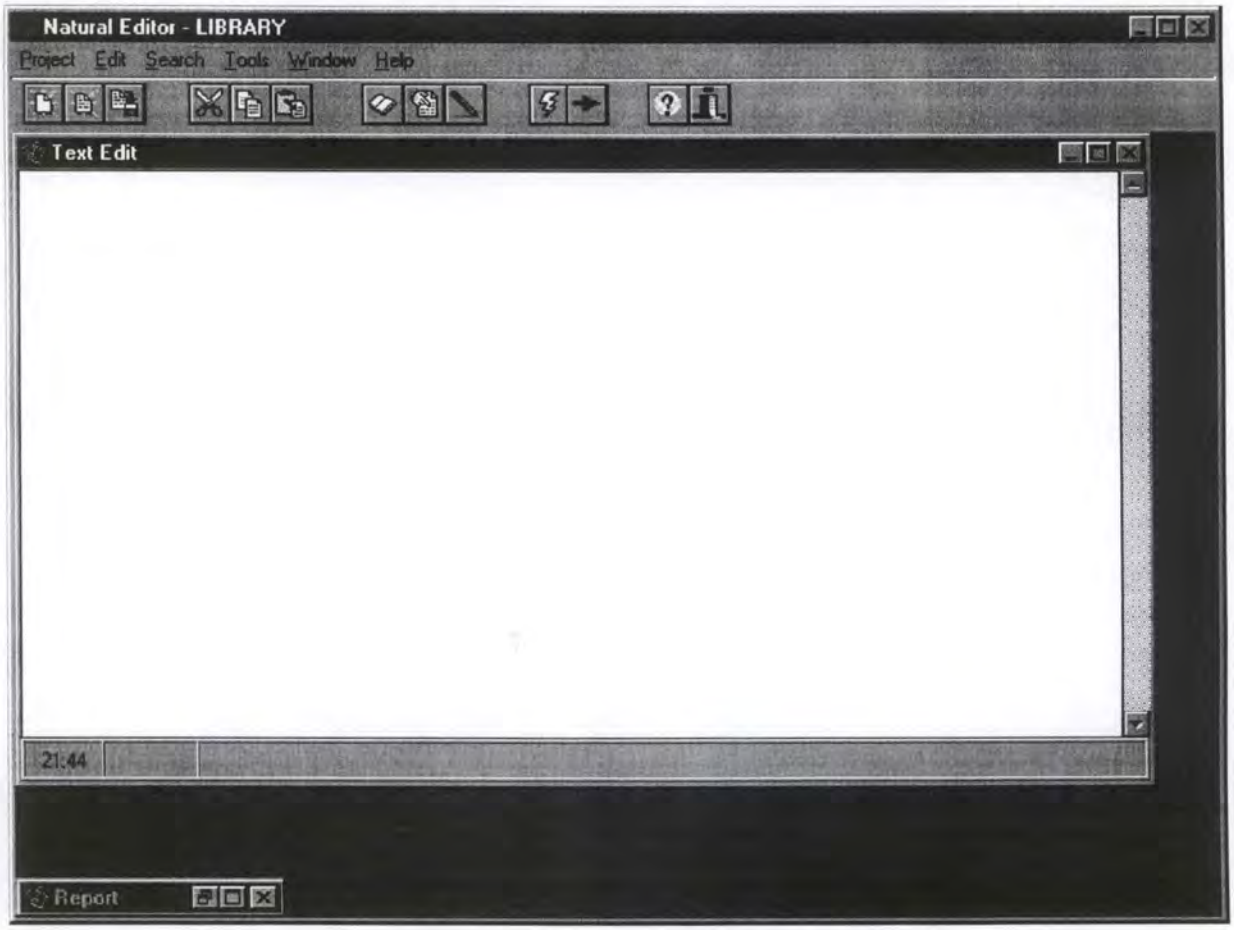


Figure 7.5. Interface de NATURAL EDITOR

Avant d'entreprendre quoique ce soit, nous allons modifier l'**environnement** de NATURAL EDITOR de façon à l'adapter à l'utilisation que nous voulons en faire. Pour ajuster les paramètres de l'environnement à notre cas, nous procédons de la manière suivante :

1. Dérouler le menu **Tools** et sélectionner **Options**. La boîte de dialogue de la figure 7.6 est affichée.
2. Activer **Stop if an error occurs during compilation**. L'activation de cette option provoquera l'arrêt de la compilation à la première erreur grammaticale ou lexicale rencontrée.
3. Dans la boîte de regroupement **Missing Verb,** cocher **Add the missing verb to the lexicon after confirmation**. Cette option permettra à l'analyste de déterminer le sens joué par un verbe absent du lexique.
4. Dans la boîte de regroupement **Db Main**, chercher et sélectionner le fichier exécutable `DB_MAIN.EXE`.
5. Dans la boîte de regroupement **Lexicon**, chercher et sélectionner le fichier texte `LEXIQUE.TXT`.

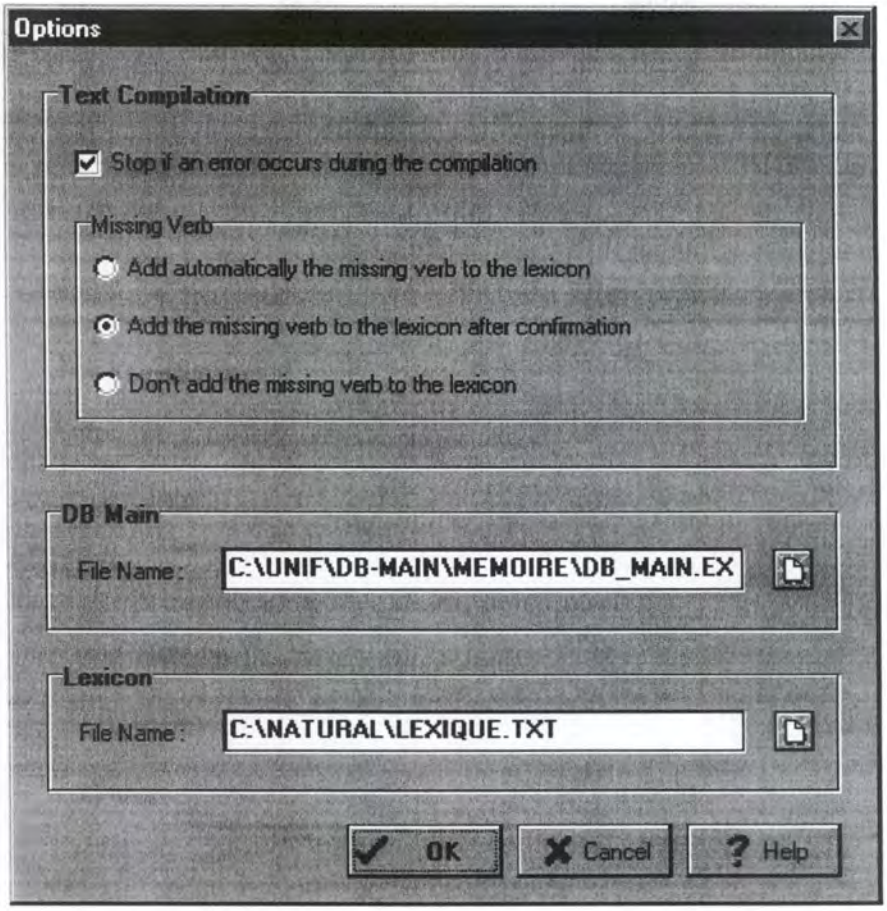


Figure 7.6. Boîte de dialogue **Options**

7.4.1.2 Saisie du texte

Nous sommes maintenant prêts à entrer notre texte dans **Text Edit.** Par souci de clarté, nous structurons notre texte en paragraphes. Nous obtenons ainsi le texte présenté à la figure 7.7.

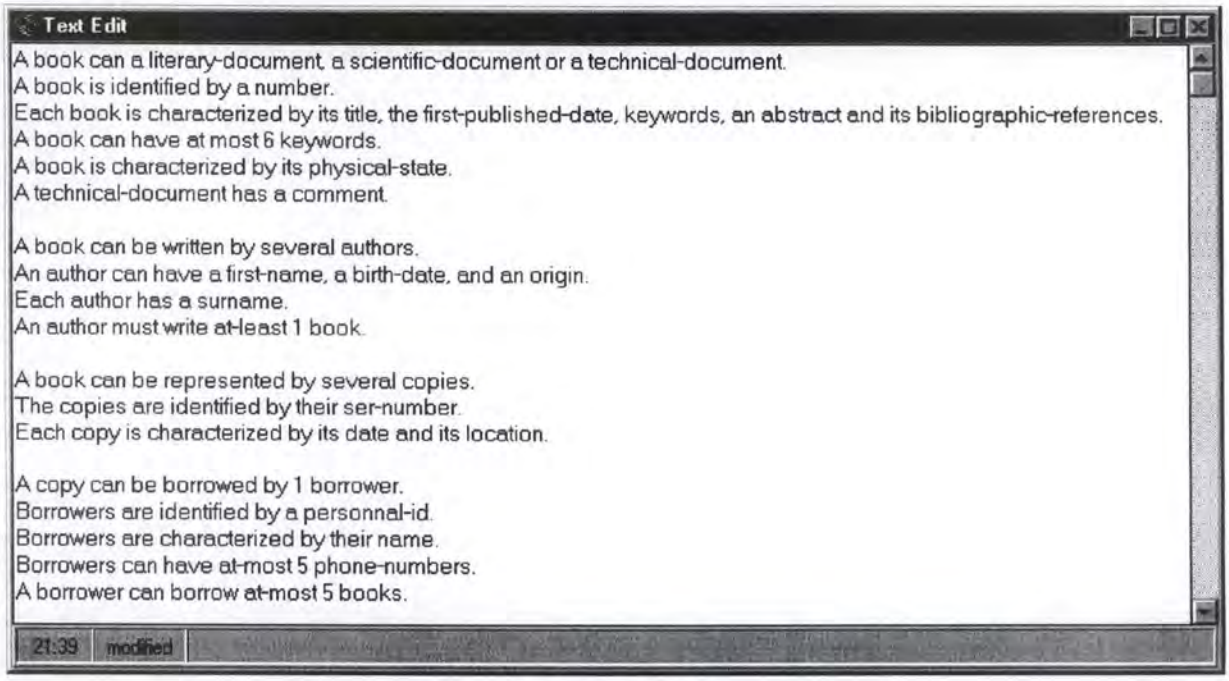


Figure 7.7. Résultat de la saisie du texte structuré

7.4.1.3 Compilation du texte

La compilation du texte a pour but de :

1. contrôler la grammaire de notre texte ;
2. ajouter interactivement les verbes absents du lexique en précisant leur sens;
3. construire le ficher **Arbre.**

Pour cette étude de cas, nous considérons que notre texte contient une erreur grammaticale à la première phrase (le groupe verbe ne contient que l'auxiliaire `can`) et que les verbes liés au domaine d'application ne sont pas présents dans le lexique.

Pour procéder à la compilation, il faut dérouler le menu **Tools** et sélectionner **Compile Text Edit**. Une barre de progression informe l'état d'avancement de la compilation.

La compilation s'arrête : une **erreur grammaticale** a été rencontrée. La boîte de dialogue de la figure 7.8 est affichée. Elle informe l'analyste que la phrase de la première ligne est grammaticalement incorrecte (la compilation s'est arrêtée à la première ligne).

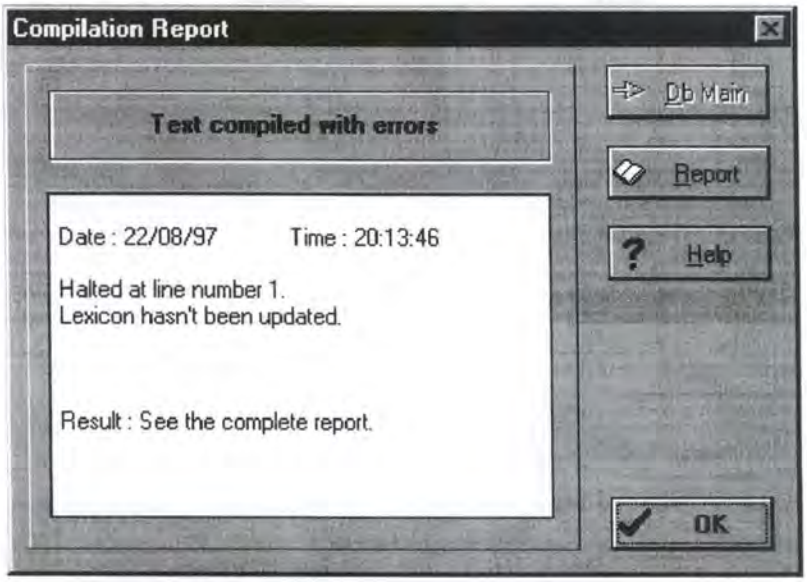


Figure 7.8. Rapport de compilation

Pour connaître la nature de l'erreur grammaticale que nous avons commise, nous consultons le rapport présenté dans la boîte de dialogue **Report** (cfr. figure 7.9). Il renseigne que le verbe a été oublié.

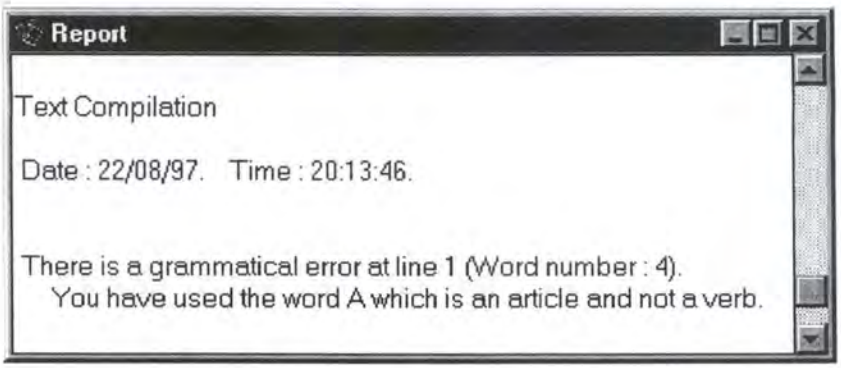


Figure 7.9. Identification de l'erreur grammaticale

Nous corrigeons la phrase en insérant le verbe `be` entre le mot `can` et `a`. Nous relançons ensuite la compilation (le texte est maintenant supposé sans erreur grammaticale).

Pour chaque **verbe absent** du lexique, la boîte de dialogue **Add a missing verb to the lexicon** est affichée (cfr. figure 7.10). Elle permet à l'analyste d'ajouter le verbe absent et d'en préciser le sens et le substantif (c'est-à-dire sa forme primitive, cfr. chapitre 5, paragraphe 5.2.1).

Trois verbes sont absents du lexique de base (`write, borrow, represent`). Il s'agit de verbes étroitement liés au domaine d'application (gestion d'une bibliothèque) et décrivant une relation entre deux objets du monde réel. Par conséquent, nous les rangeons dans la catégorie des verbes d'**action** et nous les ajoutons dans le lexique (bouton **Add**).

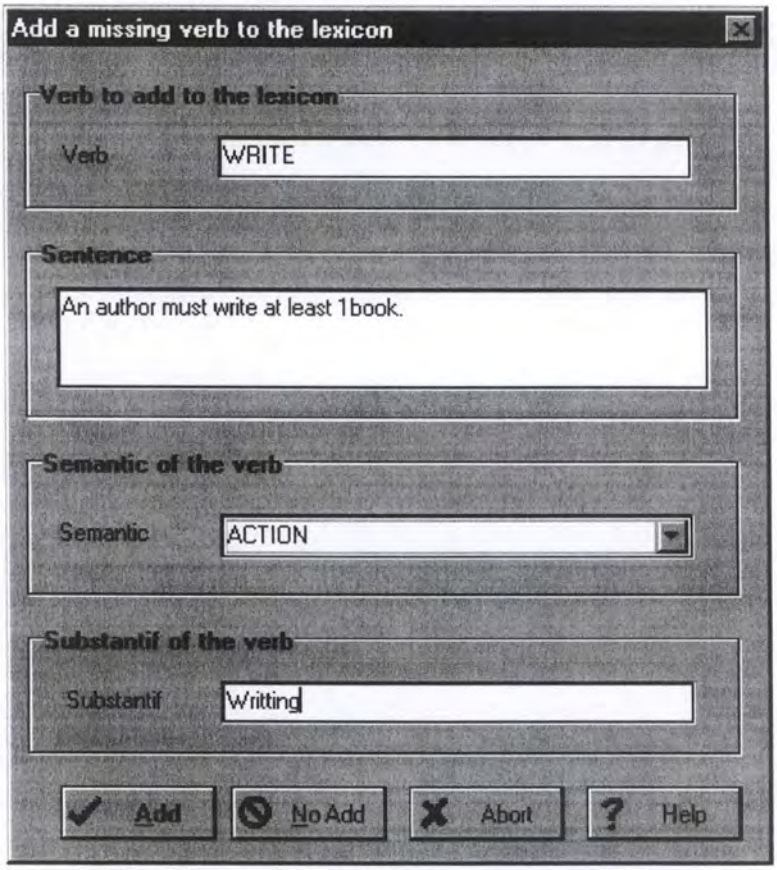


Figure 7.10. Ajout d'un verbe dans le lexique

La terminaison correcte de la compilation est indiquée par la boîte de dialogue de la figure 7.11. A partir de cette boîte de dialogue, nous pouvons :

1. visualiser le rapport complet contenant l'historique de la compilation (en particulier, les verbes ajoutés dans le lexique) ;
2. continuer la conception du schéma sémantique élémentaire en lançant l'application DB-MAIN.

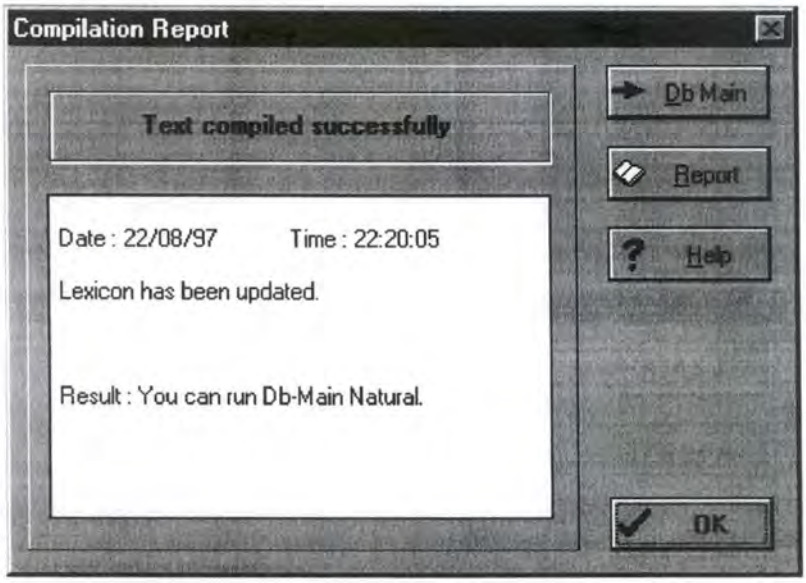


Figure 7.11. Rapport de la compilation

7.4.2 Phases de reconnaissance et d'interprétation

Une fois DB-MAIN lancé, nous pouvons exécuter le programme NATURAL DB-MAIN I. Le nom de fichier de ce programme est **N1.OXO**. Pour ouvrir NATURAL DB-MAIN I, nous procédons de la façon suivante :

1. Dérouler le menu **File** et sélectionner **Execute Voyager**. La boîte de dialogue **Load a Voyager Program** (cfr. figure 7.12) est affichée.
2. Chercher et sélectionner le fichier **N1.OXO**.
3. Cliquer sur le bouton **OK** pour lancer l'exécution de NATURAL DB-MAIN I.

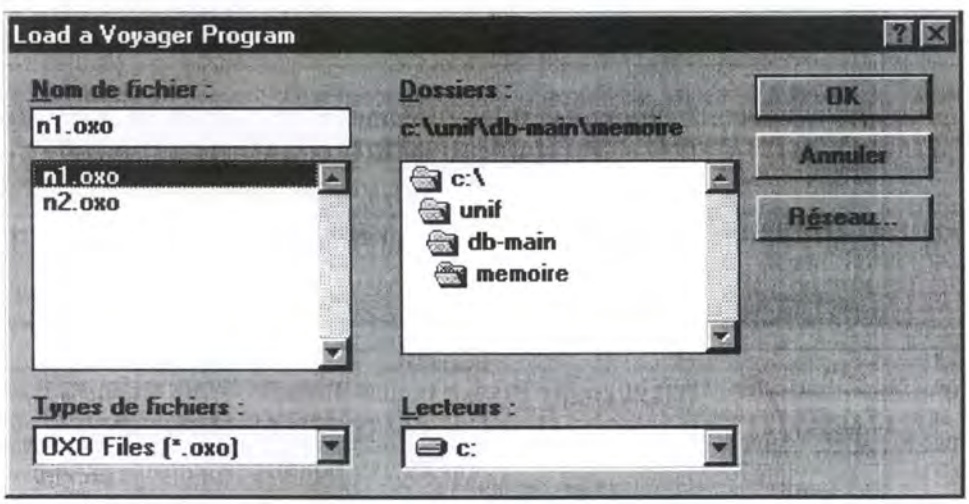


Figure 7.12. Boîte de dialogue **Load a Voyager Program**

La **console Voyager** (cfr. figure 7.13) informe sur la progression de l'exécution de NATURAL DB-MAIN I. Elle affiche les principales actions effectuées :

1. création du projet `LIBRARY` ;
2. création du schéma `LIBRARY/Elementary` (le futur schéma sémantique élémentaire) ;
3. construction progressive du schéma sémantique élémentaire par insertions de types d'entité, de types d'association, etc.

La terminaison correcte du programme est indiquée par le message suivant : `End of Interpretation Phase.`

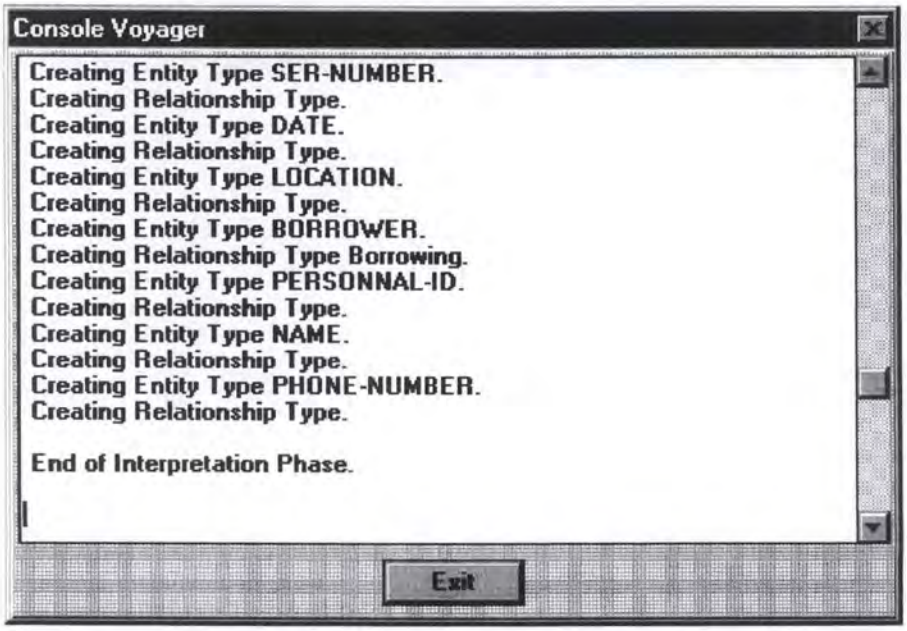


Figure 7.13. Console Voyager

Le schéma `LIBRARY/Elementary` contient le schéma sémantique élémentaire résultant de l'exécution de NATURAL DB-MAIN I (cfr. figure 7.14).

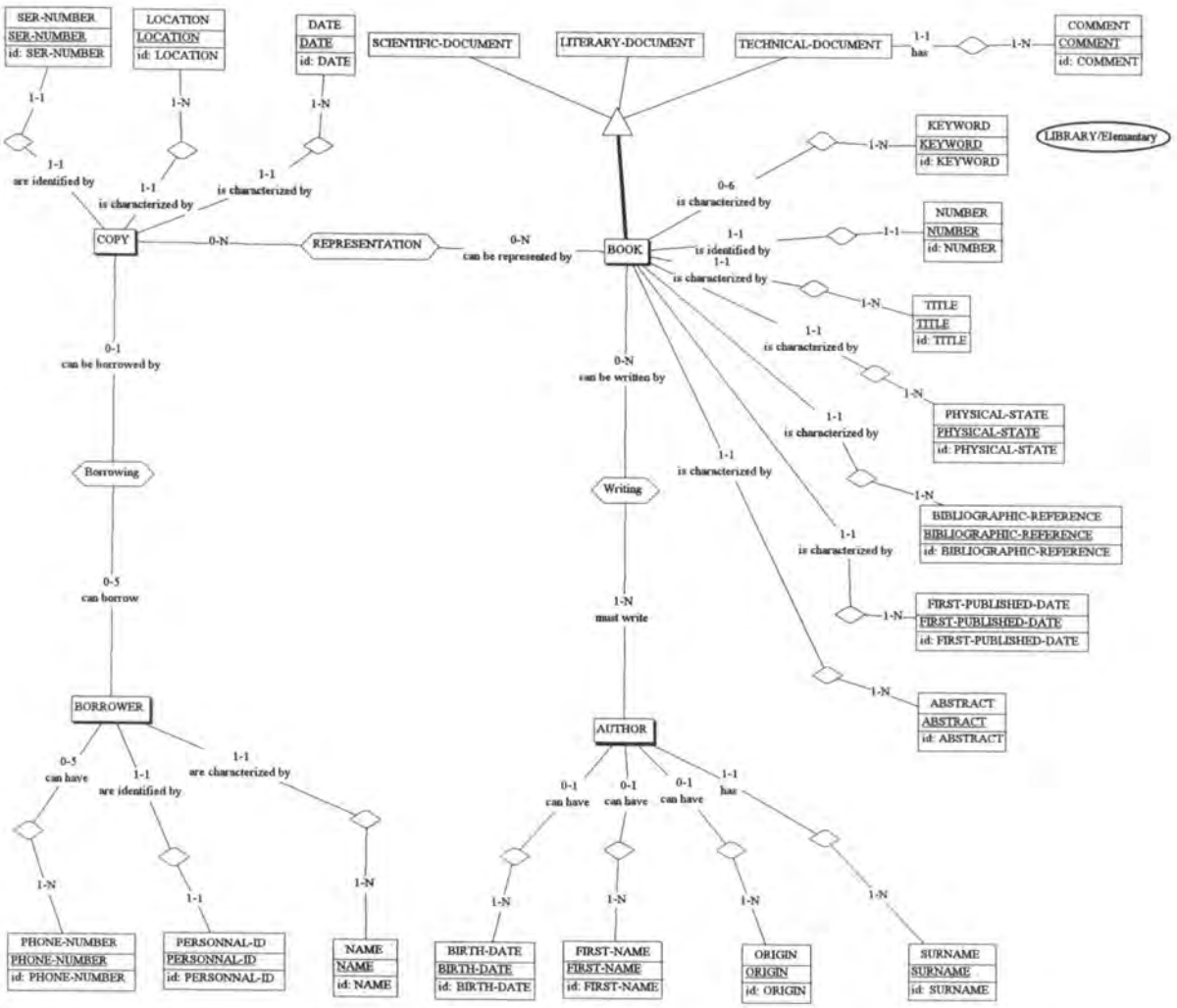


Figure 7.14. Schéma sémantique élémentaire

7.5 Troisième étape : transformation du schéma sémantique élémentaire en schéma EA

La transformation du schéma sémantique élémentaire est exécutée automatiquement par l'outil case DB-MAIN. Le résultat de cette transformation est un schéma Entité-Association de base (cfr. figure 7.15).

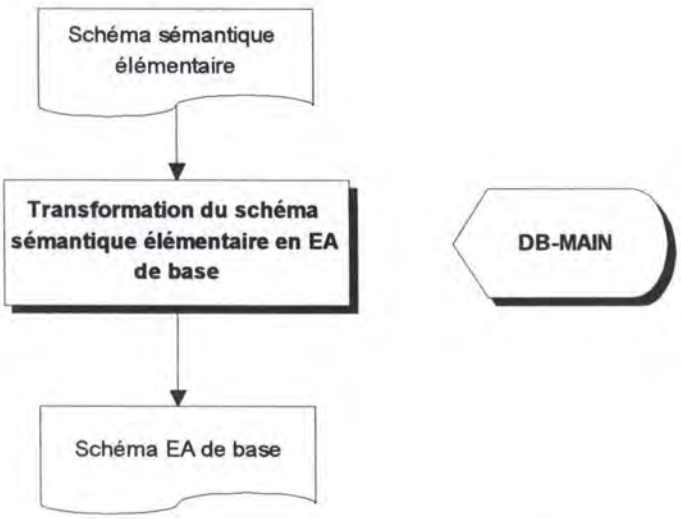


Figure 7.15. Transformation du schéma sémantique élémentaire supportée par DB-MAIN

Avant de transformer le schéma sémantique élémentaire, nous procédons à sa copie afin d'en conserver la trace. Pour copier un schéma, la procédure DB-MAIN est la suivante :

1. Dérouler le menu **Product** et sélectionner **Copy Schema**. La boîte de dialogue **Schema Properties** est affichée. Elle propose les caractéristiques par défaut du schéma : le nom du schéma est le nom du projet tandis que la version affichée est `Elementary-1`.
2. Dans la zone **Version**, remplacer `Elementary-1` par `Entity-Relation`.
3. Cliquer sur le bouton **OK** pour valider l'opération.

Nous pouvons maintenant transformer le schéma intitulé `LIBRARY/Entity-Relation`. DB-MAIN propose un assistant de transformation globale, le **Global Transformations**. La procédure DB-MAIN est la suivante :

1. Dérouler le menu **Assist** et sélectionner **Global Transformations**. La boîte de dialogue **Global Transformations** est affichée.
2. Dans la zone **Entity Types**, sélectionner **Att. Entity Types**. La zone **into** propose la seule transformation possible : **Attributes**.
3. Ajouter la transformation dans le **script** en cliquant sur **ADD**.
4. Dans la zone **Entity Types**, sélectionner **Rel. Entity Types**. La zone **into** propose la seule transformation possible : **Rel-types**.
5. Ajouter la transformation dans le **script** en cliquant sur **ADD**. La figure 7.14 montre le résultat obtenu.

6. Cliquer sur le bouton **OK** pour procéder à la transformation automatique.

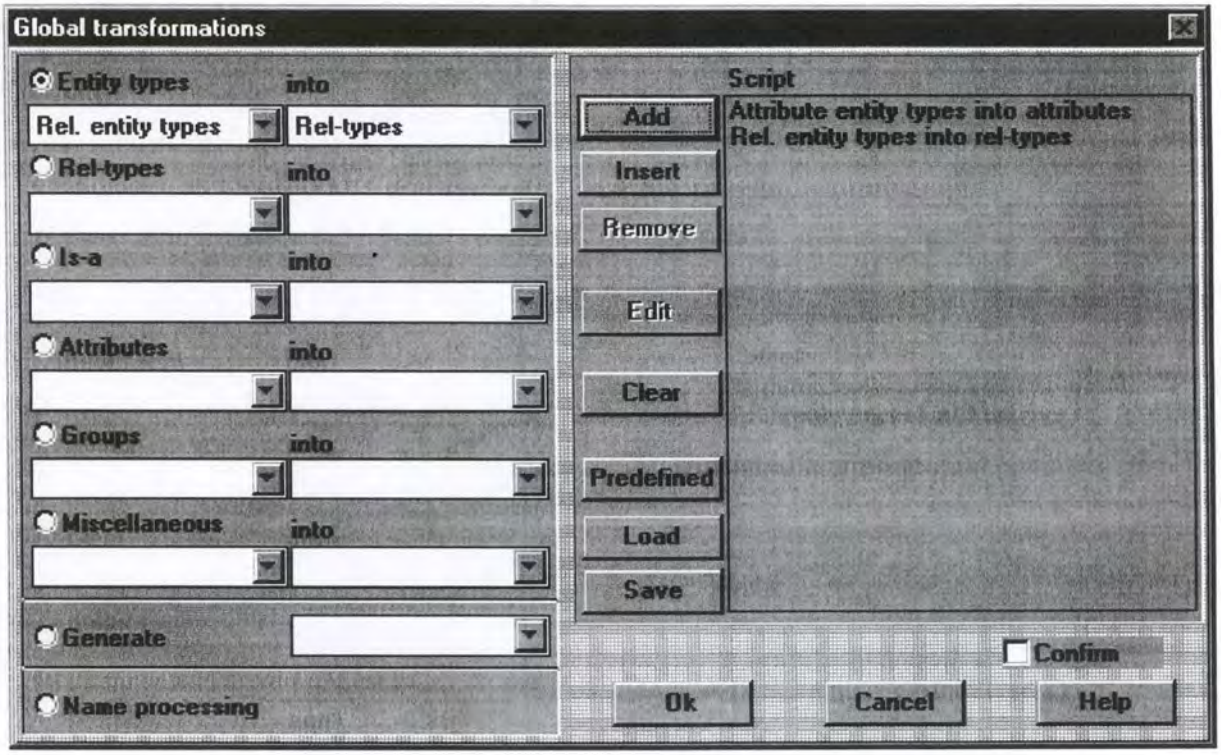


Figure 7.16. Boîte de dialogue **Global Transformations**

La procédure de transformation aboutit à un schéma EA de base (figure 7.15) mettant en évidence quatre concepts : `Author`, `Book`, `Borrower` et `Copy`.

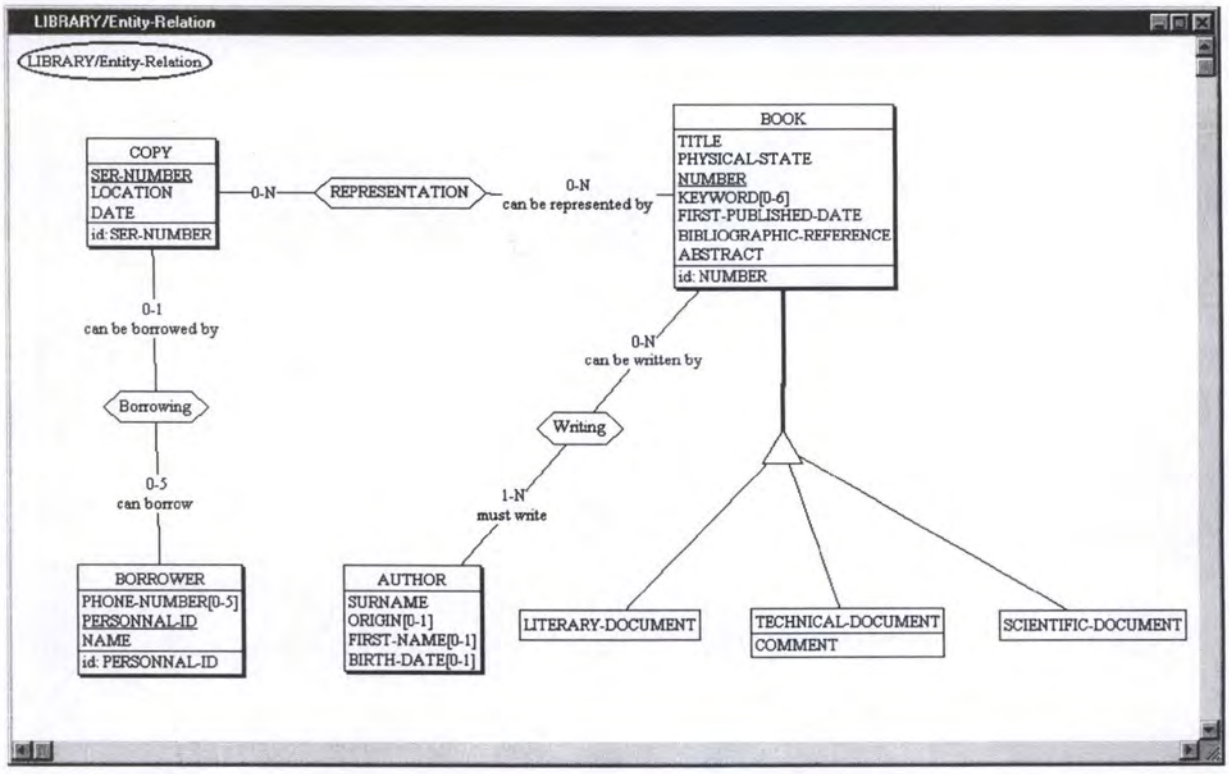


Figure 7.17. Schéma Entité-Association de base

7.6 Quatrième étape : validation du schéma

La phase de validation utilise les règles de validation établies au chapitre 3 et destinées à vérifier si les spécifications obtenues dans le modèle EA de base sont correctes (validation formelle) et à s'assurer que le schéma est bien la représentation fidèle du domaine d'application (validation du contenu).

7.6.1 Validation formelle

La validation formelle du schéma EA de base est obtenue par application des règles formelles définies au chapitre 3, paragraphe 3.5.1. Il s'agit de détecter les TE sans attribut, les TE sans identifiant, les TA similaires et les cardinalités indéterminées.

La validation formelle est entièrement supportée par NATURAL DB-MAIN II qui applique ces règles et produit un rapport contenant les structures à problème présentes dans le texte. Ce rapport est communiqué à l'analyste via NATURAL EDITOR.

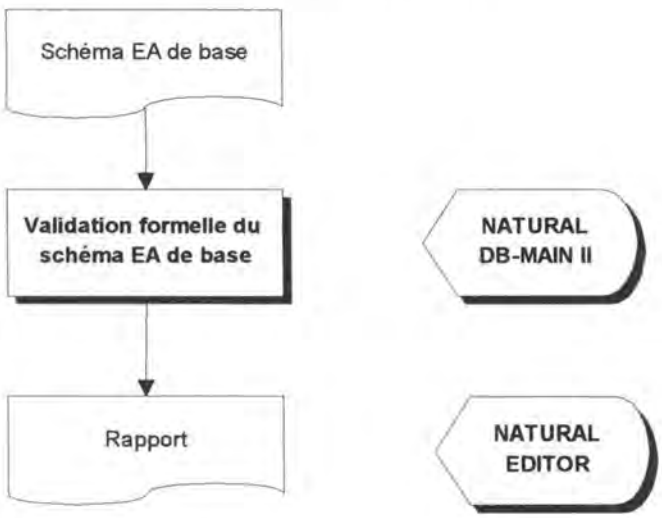


Figure 7.18. Validation formelle supportée par nos outils logiciels

Pour détecter les structures à problème présentes dans le schéma EA de base, il faut exécuter le programme NATURAL DB-MAIN II. Le nom de fichier de ce programme est **N2.OXO**. Une fois NATURAL DB-MAIN II lancé, la console **Voyager** est affichée. Elle indique la fin d'exécution par le message `End of Validation Phase`.

Le rapport (cfr. figure 7.17) fait apparaître un certain nombre de lacunes dans la spécification de départ : le type d'entité `Author` n'a pas d'identifiant et le rôle joué par `Copy` dans le cadre de l'association `Representation` n'a pas été spécifié. Ce rapport peut servir de base à un nouvel entretien avec les employés de la bibliothèque.

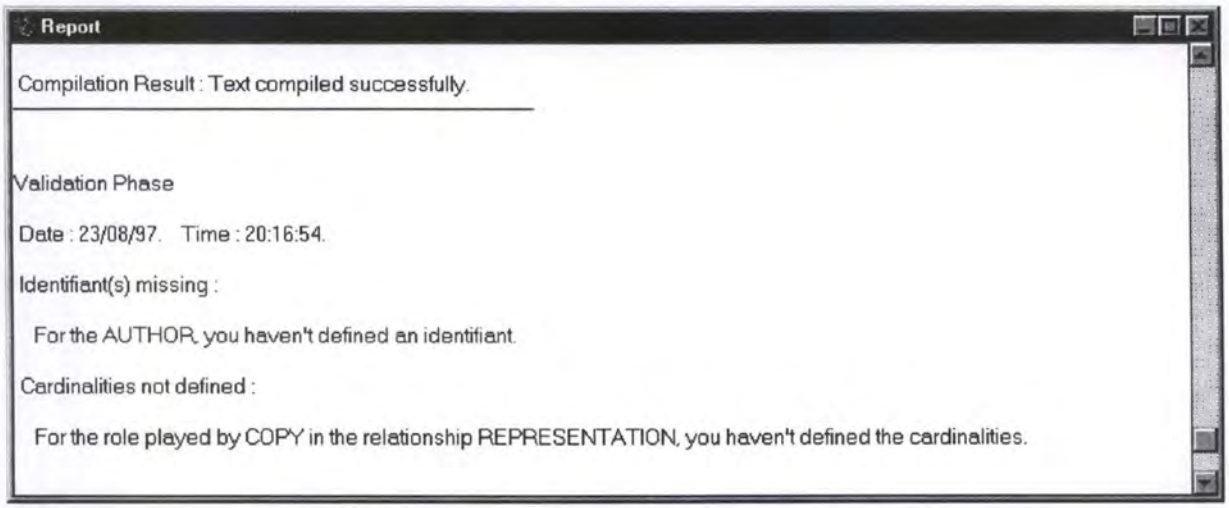


Figure 7.19. Rapport présentant les structures à problème

7.6.2 Validation du contenu

La validation du contenu consiste principalement à s'assurer que l'information contenue dans le schéma EA de base est une représentation fidèle et complète du domaine d'application. Elle doit permettre aux employés de la bibliothèque de vérifier l'adéquation du schéma conceptuel avec leur représentation du domaine d'application (cfr. chapitre 3, paragraphe 3.5.2).

Conclusions

Ce mémoire a présenté une **méthode** de conception d'un schéma conceptuel de données à partir du **langage naturel**. Nous avons caractérisé notre méthode de conception par trois composants de base une **démarche** fondée sur des **modèles** et mise en œuvre à l'aide **d'outils logiciels**.

Notre méthode s'appuie sur deux **modèles** conceptuels : le modèle Entité-Association de base et le modèle sémantique élémentaire. Le modèle Entité-Association est actuellement un des modèles conceptuels le plus utilisé. Le modèle sémantique élémentaire permet d'exprimer le contenu des textes dans une forme proche du langage naturel. Il est un intermédiaire entre le langage naturel et le schéma EA de base.

La **démarche** proposée dans ce mémoire est formée d'étapes en vue de maîtriser le processus de conception d'un schéma conceptuel à partir d'un énoncé exprimé en langage naturel. Nous avons caractérisé notre démarche comme un enchaînement de quatre étapes élémentaires :

1. **une étape d'analyse** (décomposition de l'énoncé sous forme d'un texte structuré de phrases élémentaires) ;
2. **une étape d'élaboration** (construction du schéma sémantique élémentaire à partir des éléments du texte structuré) ;
3. **une étape de transformation** (transformation du schéma sémantique élémentaire en schéma EA de base) ;
4. **une étape de validation** (détection des structures à problème présentes dans le schéma EA de base et validation du contenu).

Les **outils logiciels** que nous avons développés supportent notre démarche. Ils ont pour vocation d'assister l'analyste durant tout le processus de conception du modèle conceptuel. Cette aide intervient durant la phase d'élaboration du schéma sémantique élémentaire, durant la phase de transformation du schéma sémantique élémentaire en objets du modèle sémantique élémentaire et enfin durant la phase de validation du schéma.

L'élaboration du schéma sémantique élémentaire consiste à associer les objets du monde réel aux concepts du modèle. Le **rôle du langage** dans cette activité d'abstraction a orienté notre mémoire vers un formalisme des **mécanismes linguistiques** utilisés par l'analyste pour traduire les éléments d'une phrase élémentaire en concepts. Ces mécanismes sont fondés sur la reconnaissance de structures de phrases familières. L'interprétation de ces structures permet à l'analyste d'associer aux faits de la phrase des concepts du modèle sémantique.

L'approche linguistique mise en œuvre dans nos outils logiciels est empruntée à la théorie de Fillmore. Fillmore exprime l'idée que le sens d'une phrase est dérivable du sens du verbe et des rôles imbriqués. Ceci nous a conduits à identifier un ensemble de structures de phrases permettant de déduire le sens d'une phrase en fonction de sa structure.

Nous avons défini un ensemble de rôles sémantiques spécifiques à la construction d'un schéma sémantique élémentaire. Partant de ces rôles, nous avons retenu quatre classes de verbes : les verbes de description, les verbes d'identification, les verbes de spécialisation et les verbes

d'action . Nous avons ensuite défini un ensemble de schémas qui combinent les rôles et les classes de verbes précédemment introduites. Ces schémas représentent des structures de phrases simples. Ils sont de deux types : schémas structuraux et schémas associatifs.

Nous avons ensuite mis en œuvre l'approche linguistique pour générer automatiquement un schéma sémantique élémentaire à partir d'un texte structuré de phrases élémentaires. L'élaboration du schéma sémantique élémentaire repose sur un ensemble de règles qui mettent en œuvre les mécanismes linguistiques utilisés par l'analyste. Ces règles de traduction sont de trois types : règles lexico-syntaxiques, règles de caractérisation et règles d'interprétation.

Elles sont utilisées de façon à transformer les énoncés en langage naturel en schéma sémantique élémentaire. Le processus de transformation peut être vu comme un déroulement de trois phases.

La **phase de représentation** consiste à construire une représentation interne des phrases sous forme d'arbres syntaxiques mettant en évidence leur structure grammaticale. Cette phase applique des règles lexicales et syntaxiques. Le rôle des règles lexicales est de déterminer la nature grammaticale de chacun des mots d'une phrase et d'affecter le verbe à l'une des quatre classes. Ces règles lexicales utilisent un lexique qui contient des informations sur la nature grammaticale des mots et sur la classification des verbes. Les règles syntaxiques permettent d'une part de vérifier que la phrase est conforme au langage utilisé et, d'autre part, de construire les arbres syntaxiques. Ces règles reposent sur une grammaire générative qui corresponde à la connaissance grammaticale de nos outils logiciels.

La **phase de reconnaissance** cherche à unifier les arbres syntaxiques aux schémas linguistiques définis de manière à reconnaître le rôle de chacun des mots de la phrase. La reconnaissance du schéma approprié à une phrase et l'association des rôles aux propositions sont effectuées par la mise en œuvre des règles de caractérisation. La reconnaissance du schéma est fondée sur la classe de verbe (identifiée au cours de la première étape et attachée à l'arbre) et sur la structure grammaticale de la phrase.

Enfin, **l'étape d'interprétation** assure la construction du schéma sémantique élémentaire. Elle se fonde sur l'ensemble de règles traduisant la correspondance entre rôle sémantique et concepts du modèle. Ces règles permettent la construction automatique du schéma sémantique à partir de la reconnaissance des schémas et des rôles établie à l'étape précédente.

L'utilisation de l'approche linguistique pour formaliser la phase d'abstraction, point de départ de toute activité de conception, est intéressante ; c'est un des moyens de comprendre et de formaliser le processus d'abstraction. Même si actuellement il semble improbable de pouvoir utiliser nos outils logiciels pour donner l'ensemble de spécifications d'un schéma conceptuel, il est raisonnable de les utiliser de manière à construire un premier schéma conceptuel constituant une base de travail sur laquelle l'analyste peut construire un schéma conceptuel complet et cohérent.

La typologie des rôles sémantiques et des structures de phrases peut être améliorée et approfondie. Nous estimons qu'elle mérite de l'être. L'approche linguistique donne un éclairage nouveau aux modèles conceptuels ([ROLLAND, 91]). Ceci nous semble intéressant dans l'optique de supporter les analystes dans leurs tâches d'abstraction des situations réelles auxquelles ils sont confrontés.

Concrètement, l'approche linguistique a permis de doter l'atelier DB-MAIN d'une interface en langage naturel, complémentaire de l'interface graphique et permettant à l'atelier DB-MAIN de démarrer le processus de conception d'une base de données, non pas en partant d'un schéma conceptuel qui suppose la tâche d'abstraction déjà réalisée, mais en partant d'une description du domaine d'application exprimée en langage naturel. Ce qui met son utilisation à la portée d'un nombre plus important de personnes. Il convient cependant de rester prudent quant à sa protée. En effet, la tâche la plus difficile à mener reste la représentation du contenu informationnel sous la forme d'un texte structuré de phrases élémentaires.

Finalement, nous pensons que nos outils logiciels peuvent jouer un rôle pédagogique auprès d'analystes débutants et d'étudiants en leur proposant un environnement méthodologique interactif basé sur une approche linguistique.

Bibliographie

[ABRIAL, 74] Abrial J.R., *« Data Semantics in Data Management Systems »*, Kimbic and Koffeman (eds), North Holland, 1974

[ALBRECHT, 96] Albrecht M., Thalheim B. (Eds), *« Challenges of Application and Challenges of Design »*, Proceedings of the Workshops, 15th International Conference on Conceptual Modeling, ER'96, Brandenburg Technical University, Cottbus, Allemagne, 1996

[AMBROSIO, 95] Ambrosio A. P., Métais E., Meunier J-N., *« The Linguistic Level of the KHEOPS Case Tool »*, Laboratoire PRiSM, Université de Versailles, France, 1995

[ANSI, 77] ANSI/X3/SPARC, *« Interim Report on Data Base Systems »*, 1977

[BATINI, 92] Batini C., Ceri S, Navathe S. B., *« Conceptual Database Design : an Entity-Relationship Approach »*, The Benjamin/Cummings Publishing Company, Redwood City, Etats-Unis, 1992

[BODART, 94] Bodart F., Pigneur Y., *« Conception assistée des systèmes d'information. Méthode, modèles, outils »*, Masson, Paris, 1994

[BUCHHOLZ, 96] Buchholz E., Cyriaks H., Düsterhöft A., Mehlan H., Thalheim B., *« Applying a Natural Langage Dialogue Tool for Designing Databases »*, in [ALBERCHT, 96]

[CHEN, 76] Chen P. P., *« The Entity-Relationship Model : Toward a Unified View of Data »*, ACM TODS, volume 1, n°1, 1976

[CHOMSKY, 65] Chomsky N., *« Aspects of the Theory of Syntax »*, MIT Press, Cambrige, Massachusetts, Etats-Unis, 1965

[CLAUSS, 96] Clauss W., Thalheim B. (Eds), *« ER CASE Tools -Industrial Track- »*, Proceedings of the Workshop, 15th International Conference on Conceptual Modeling, ER'96, Brandenburg Technical University, Cottbus, Allemagne, 1996

[COOD, 70] Cood E.F., *« A Relationnal Model of Data for Large Shared Data Banks »*, Comm. ACM, volume 13, n°6, 1970

[DALIANIS , 92] Dalianis H., *« A Method for Valiting a Conceptual Model by Natural Language Discourses Generation »*, CAISE-92, International Conference on Advanced Information Systems Engineering, Ed. Loucopoulos P., Springer, 1992

[DALIANIS, 96] Dalianis H., *« Explaining Conceptual Models - An architecture and Design Principles »*, Department of Computer and Systems Sciences, Université de Stockholm, Suède, 1996

[DB-MAIN, 95] *« DB-MAIN Tutorial. Volume 1 : Introduction to Database Design »*, DB-Main Project, Institut d'informatique, FUNDP, Namur, 1995

[DEFLORENNE, 96] Deflorenne A., *« Natural : un paraphraseur de schéma conceptuel de bases de données. Manuel de référence »*, DB-Main Project, Institut d'informatique, FUNDP, Namur, 1996

[DELOBEL, 91] Delobel C., Lécluse C., Richard, *« Base de données : du systèmes relationnels aux systèmes à objets »*, InterEditions, Paris, 1991

[ENGLEBERT, 97] Englebert V., *« Voyager II, Reference Manual »*, Institut d'informatique, FUNDP, Namur, 1997

[HAINAUT, 86] Hainaut J.-L., *« Conception assistée des applications informatiques. Conception de la base de données »*, Masson et Presses Universitaires de Namur, Paris, 1986

[HAINAUT, 94] Hainaut J.-L., *« Bases de données et modèles de calcul »*, InterEditions, Paris, 1994

[HAINAUT, 96] Hainaut J.-L., Roland D., Henrard J., Englebert C., Hick J.-M., *« DB-MAIN, A General-purpose CASE Environment for Advanced Database Applications Engineering »*, in [ALBRECHT, 96]

[HALPIN, 95] Halpin T., *« Conceptual Schema and Relational Database Design »*, Prentice Hall, Australie, 1995

[HALPIN, 96] Halpin T., *« Object-Role Modelling : an Overview »*, in [CLAUSS, 96]

[KERSTEN, 87] Kersten M.L., *« A conceptual Modedelling Expert System »*, Entity-Relationship Approach, S. Spaccapietra (Ed), Elsevier Science Publishers, North-Holland, 1987

[LOUCOPOULOS, 92] Loucopoulos P., Zicari R. (Eds), *« Conceptual Modeling, Databases and CASE : An Integrated View of Information Systems Development »*, John Wiley, New York, 1992

[OLLE, 82] Olle T.W., Sol H.G., Tully C.J (Eds), *« Information Systems Design Methodologies : A Comparative Review »*, Proccedings of the IFIP WG8.1 WC on Comparative Review of Information Systems Design Methodologies, Noordwijkerkhout, North-Holland, 1982

[PROIX, 89] Proix C., *« OICSI : un outil d'aide à la conception des systèmes d'information : spécification et réalisation »*, Thèse de doctorat de l'université de Paris VI, Paris, 1989

[ROLLAND, 88] Rolland C., Foucaut O., Benci G., *« Conception des systèmes d'information. La méthode REMORA »*, Eyrolles, Paris, 1988

[ROLLAND, 91] Rolland C., Proix C., *« Une approche linguistique pour la conceptualisation des systèmes d'informations »*, Génie Logiciel & Systèmes Experts, méthodes et outils, France, 1991

[ROLLAND, 92] Rolland C., Proix C., *« Natural Language Approach to Conceptual Modelling »*, in [Loucopoulos, 92]

[SABAH, 90a] Sabah G., *« L'intelligence artificielle et le langage. Représentation des connaissances »*, volume 1, seconde édition, Hermes, Paris, 1990

[SABAH, 90b] Sabah G., *« L'intelligence artificielle et le langage. Processus de compréhension »*, volume 2, seconde édition, Hermes, Paris, 1990

[SENKO, 73] Senko M.E., Altaman E., Astraham M., Fehder P., *« Data Structures and Accessing in Data-Base System »*, IBM sys. J, volume 12, n°1, 1973

[TARDIEU, 86] Tardieu H., Rochfeld A., Colleti R., Panet G., Vallée G., *« La méthode MERISSE : les étapes »*, Ed. D'Organisation, 1986

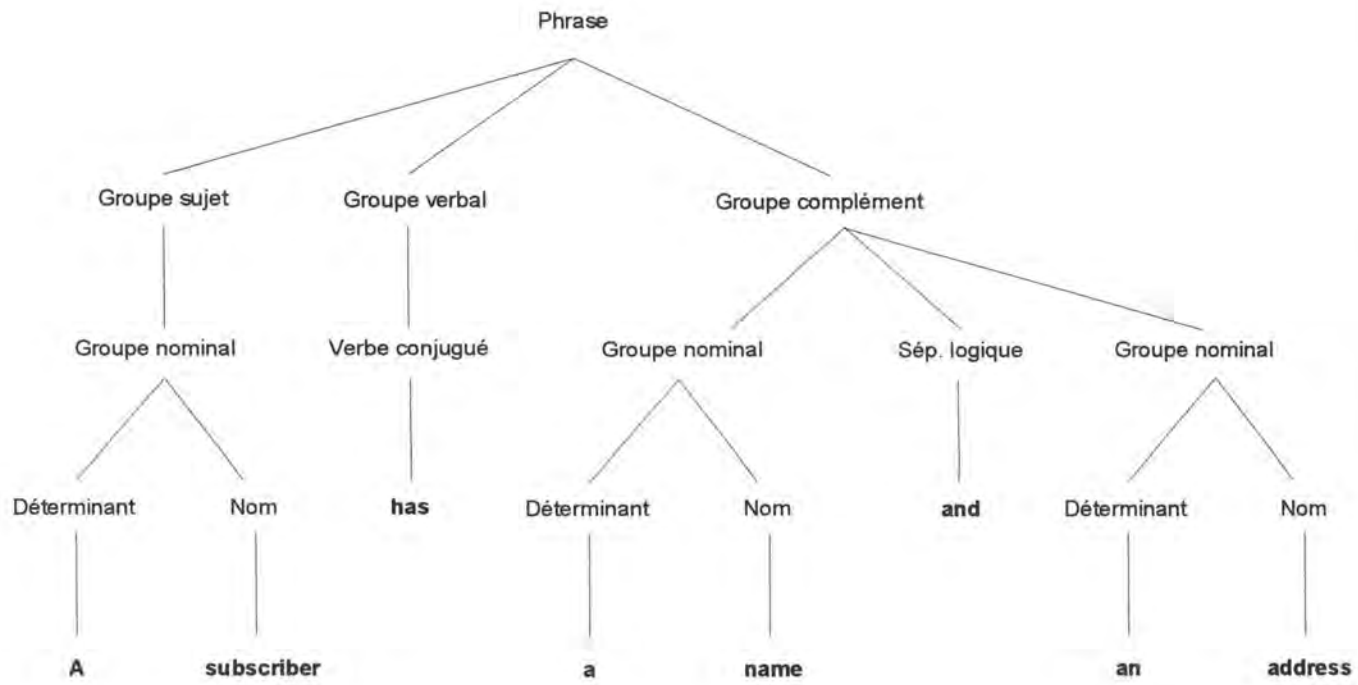
[TEOREY, 94] Teorey, T. J., *« Database Modeling & Design : The Fundamental Principles »*, seconde édition, Morgan Kaufmann Publishers, San Francisco, Etats-Unis, 1994

[VERHEIJEN, 82] Verheijen G. M. A., Van Bekkum J., *« NIAM : an Information Analysis Method »*, in [OLLE, 82]

Annexe I : Arbre syntaxique

La structure grammaticale d'une phrase est représentée par un arbre syntaxique. Les arbres syntaxiques sont des graphes dont les **noeuds** représentent les fonctions grammaticales de la phrase (c'est-à-dire les éléments appartenant au vocabulaire non terminal) et dont les **feuilles** représentent les mots appartenant au vocabulaire terminal.

Par exemple, la phrase « `A subscriber has a name and an address` » est associée à l'arbre suivant :



Annexe II : Syntaxe BNF

Conventions

Voici les conventions utilisées pour la définition de la sémantique et de la syntaxe de chaque construction du langage :

Symbole BNF	Signification
e	Symbole terminal du langage
e	Symbole non terminal du langage
::=	Définit un symbole non terminal du langage
[...]	Elément facultatif
{...}	Elément qui peut être répété de 0 à N fois
...|...|...	Choix d'un élément parmi N

Symboles spéciaux

Voici la liste des symboles spéciaux utilisés :

Symbole BNF	Signification
SautLigne	Symbole représentant la fin d'une ligne (EOF)
Espace	Symbole représentant l'espace vide d'un caractère.

Types

Le langage supporte deux types de valeurs : numérique et chaîne de caractères. Pour chacun de ces deux types, nous précisions quel est l'ensemble des valeurs qui constitue ce types.

Type numérique

L'ensemble des valeurs est l'ensemble des nombres entiers positifs ou nuls. Pratiquement, il n'est pas possible de représenter tous les nombres entiers : nous nous contenterons des entiers compris entre un certain intervalle ([0,1000] par exemple).

Syntaxe BNF pour la construction des nombres :

```
Nombre  ::= Chiffre{Chiffre}
Chiffre ::= 1|2|3|4|5|6|7|8|9|0
```

Type alphabétique

L'ensemble des valeurs est l'ensemble formés de caractères alphabétiques et des caractères apostrophe (') et trait d'union (-).

Syntaxe BNF pour la construction des mots :

```
ChaîneCar   ::= Lettre{Caractères}
Caractère   ::= Lettre|CarSpécial
Lettre      ::= LettreMaj|LettreMin
LettreMaj   ::= A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|
                V|W|X|Y|Z
LettreMin   ::= a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|
                v|w|x|y|z
CarSpécial  ::= -|'
```

Annexe III : Fichier Arbre

Description

Le fichier **Arbre** (`NomProjet.OUT`) est un fichier texte contenant l'arbre syntaxique. Il est automatiquement généré par Natural Editor pendant la compilation du texte. Le fichier est vide si la compilation s'est terminée avec une erreur. Le fichier contient l'arbre syntaxique sinon. La structure du fichier **Arbre** respecte les règles suivantes :

1. le fichier est de format texte ;
2. une ligne est la transcription de l'arbre syntaxique associé à une phrase du texte ;
3. une ligne contient une séquence d'informations syntaxiques attachées à un mot de la phrase ;
4. une information syntaxique attachée à un mot respecte la syntaxe suivante :
    1. elle est enfermée par des parenthèses ;
    2. elle contient une séquence d'informations élémentaires : <Type de groupe> <Numéro du GN dans le groupe>,<Grammaire>,<Forme canonique>,<Forme fléchie>,<Substantif>,<Sens sémantique>, <Numéro du mot>, <Numéro de la phrase> ;
    3. les informations élémentaires sont séparées entre elles d'une virgule.

Syntaxe BNF

Un résumé complet de la syntaxe utilisée en formalisme BNF se trouve en annexe II.

```
ArbreSynt   ::= {{Ouverture InfoSynta Fermeture (Espace)}
                 SautLigne}
Ouverture   ::= (
Fermeture   ::= )
InfoSynta   ::= TypeGroupe Séparateur NuméroGN Séparateur
                NatureGram Séparateur Canonique Séparateur
                FormeFléchie Séparateur Sens Séparateur NumMot
                Séparateur NumPhrase
Séparateur  ::= ,
TypeGroupe  ::= GS|GV|GC|GI
NuméroGN    ::= Chiffre
NatureGram  ::= ARTICLE|CONJUNCTION|DISJUNCTION|INTEGER|
                MOT-CLE|PREPOSITION|VERB
Canonique   ::= ChaîneCar
FormeFléchie ::= ChaîneCar
Sens        ::= NULL|ACTION|IDENT-SUJ|IDENT-POSS|POSS-SUJ|
                POSS-COMPL|MAXIMUM|MINIMUM|OTHER
NumMot      ::= Chiffre
NumPhrase   ::= Chiffre
```

Exemple

L'exemple suivant présente la transcription de l'arbre syntaxique associé à la phrase :

« A book can have one abstract. »

Notons que la transcription de l'arbre syntaxique devrait tenir sur une ligne.

```
(GS,0,ARTICLE,A,A,A,NULL,1,4)(GS,0,WORD,BOOK,book,BOOK,NULL,2,4)(GV,0,VERB,
CAN,can,CAN,AUXILIARY,3,4)(GV,0,VERB,HAVE,have,HAVING,POSS-SUJ,4,4)
(GC,1,INTEGER,1,1,1,1,5,4)(GC,1,WORD,ABSTRACT,abstract,ABSTRACT,NULL,6,4)
```

Annexe IV : Fichier Environnement

Description

Le fichier Environnement (`NATURAL.INI`) est un fichier texte présent dans le répertoire `C:\WINDOWS`. Il est automatiquement généré par NATURAL EDITOR et contient les informations nécessaires pour lancer NATURAL DB-MAIN I & II. Ces informations sont le nom et le chemin d'accès du projet en cours.

Syntaxe

```
[Current Project]
Directory=directory_string
Name=name_string
```

Paramètre	Syntaxe
Directory_string	Chaîne de caractères contenant le lecteur et le chemin d'accès du projet en cours
Name_string	Chaîne de caractères contenant le nom du projet en cours (sans extension)

Exemple

```
[Current Project]
Directory=C:\NATURAL
Name=Library
```

Annexe V : Fichier contenant le lexique

Description

Le lexique représente la base de connaissance de NATURAL EDITOR sur le vocabulaire anglais. Pour chaque mot qu'il comprend, le lexique mémorise quatre types d'informations :

1. la forme canonique du mot, c'est-à-dire son entrée dans le dictionnaire ;
2. un mot apparenté (si le mot est un verbe, alors le mot apparenté correspond au substantif de ce verbe) ;
3. la nature grammaticale du mot ;
4. si le mot est un VERB, le sens ou le type du verbe : AUXILIARY TO BE, AUXILIARY, ACTION, IDENT-SUJ, IDENT-POSS, POSS-SUJ ou POSS-COMPL. Si le mot est une INFORMATION, la classe dans laquelle elle appartient : MINIMUM, MAXIMUM ou OTHER. Pour un autre type de mot, la valeur NULL.

Syntaxe BNF

Un résumé complet de la syntaxe utilisée en formalisme BNF se trouve en annexe II.

```
Lexique     ::= Contenu SautLigne {Contenu SautLinge}
Contenu     ::= Canonique Séparateur [Substantif] Séparateur
                NatureGram Séparateur Sens
Séparateur  ::= ,
Canonique   ::= ChaîneCar
Substantif  ::= ChaîneCar
NatureGram  ::= ARTICLE|CONJUNCTION|DISJUNCTION|INTEGER|
                MOT-CLE|PREPOSITION|VERB
Sens        ::= NULL|ACTION|IDENT-SUJ|IDENT-POSS|MINIMUM|
                MAXIMUM|OTHER|POSS-SUJ|POSS-COMPL
```

Exemple

```
BORROW,BORROWING,VERB,ACTION
THE,,ARTICLE,NULL
MAXIMUM,,MOT-CLE,MAXIMUM
```

Annexe VI : Représentation interne de l'arbre syntaxique

Les arbres syntaxiques sont stockés dans NATURAL DB-MAIN au moyen de listes dont la structure est la suivante :

Le `Texte` possède la représentation interne :

```
[Phrase_1, Phrase_2, ..., Phrase_n]
```

où `Phrase_i` est la liste des phrases appartenant au texte.

Une `Phrase_i` quelconque possède la représentation interne :

```
[GroupeSujet, GroupeVerbal, GroupeInformatif,
GroupeComplément, TextePhrase, TexteVerbe]
```

Le `GroupeSujet` possède la représentation interne :

```
[Mot_1, Mot_2]
```

où `Mot_i` est une liste correspondant à un mot appartenant au groupe sujet (le groupe sujet peut contenir au maximum 2 mots : l'article et le nom).

Un `Mot_i` possède la représentation interne

```
[NatureGrammaticale, Canonique, MotReel, Position]
```

où
- `NatureGrammaticale` est une chaîne de caractères représentant la nature grammaticale du mot ;
- `Canonique` est une chaîne de caractères représentant la forme canonique du mot ;
- `MotReel` est une chaîne de caractères représentant le mot de la phrase;
- `Position` est un entier représentant la position du mot dans la phrase.

Le `GroupeVerbal` possède la représentation interne :

```
[Verbe_1, Verbe_2, Verbe_3, Preposition]
```

où
- `Verbe_i` est une liste correspondant à un verbe appartenant au groupe verbal (le groupe verbal peut contenir au maximum quatre mots : l'auxiliaire de mode, l'auxiliaire être, le verbe conjugué et une préposition) ;
- `Preposition` est une liste correspondant à la préposition appartenant au groupe verbal.

Un `Verbe_i` possède la représentation interne

```
[NatureGrammaticale, Canonique, MotReel, Substantif,
Sémantique, Position]
```

où
- `NatureGrammaticale` est une chaîne de caractères représentant la nature grammaticale du verbe ;
- `Canonique` est une chaîne de caractères représentant l'infinitif du verbe ;
- `MotReel` est une chaîne de caractères représentant le verbe conjugué ;
- `Substantif` est une chaîne de caractères représentant le substantif du verbe ;
- `Sémantique` est une chaîne de caractères représentant le sens du verbe ;
- `Position` est un entier représentant la position du verbe dans la phrase.

Une `Preposition` possède la représentation interne

```
[NatureGrammaticale, Canonique, MotReel, Position]
```

où
- `NatureGrammaticale` est une chaîne de caractères représentant la nature grammaticale du mot ;
- `Canonique` est une chaîne de caractères représentant la forme canonique du mot ;
- `MotReel` est une chaîne de caractères représentant le mot de la phrase;
- `Position` est un entier représentant la position du mot dans la phrase.

Le `GroupeInformatif` possède la représentation interne :

```
[Info_1, Info_2]
```

où `Info_i` est une liste correspondant aux mots appartenant au groupe informatif.

Une `Info_i` possède la représentation interne

```
[NatureGrammaticale, Canonique, MotReel, Position]
```

où
- `NatureGrammaticale` est une chaîne de caractères représentant la nature grammaticale du mot ;
- `Canonique` est une chaîne de caractères représentant la forme canonique du mot ;
- `MotReel` est une chaîne de caractères représentant le mot de la phrase;
- `Position` est un entier représentant la position du mot dans la phrase.

Le `GroupeComplement` possède la représentation interne :

```
[Mot_1, Mot_2, ..., Mot_n]
```

où `Mot_i` est la liste correspondant aux mots appartenant au groupe complément.

Un `Mot_i` possède la représentation interne :

`[Numéro, NatureGrammaticale, Canonique, MotRéel, Position]`

où

- `Numéro` est un entier représentant le numéro du groupe complément ;
- `NatureGrammaticale` est une chaîne de caractères représentant la nature grammaticale du mot ;
- `Canonique` est une chaîne de caractères représentant la forme canonique du mot ;
- `MotReel` est une chaîne de caractères représentant le mot de la phrase;
- `Position` est un entier représentant la position du mot dans la phrase.

Le `TextePhrase` est une chaîne de caractères représentant le texte de la phrase et le `TexteVerbe` est une chaîne de caractères représentant le texte du groupe verbal.

Annexe VII : Représentation interne des arbres syntaxico-sémantiques

Les arbres syntaxico-sémantiques sont stockés dans NATURAL DB-MAIN au moyen de listes dont la structure est la suivante :

Le `TexteSem` possède la représentation interne :

```
[PhraseSem_1, PhraseSem_2, ..., PhraseSem_n]
```

où `PhraseSem_i` est la liste des phrases appartenant au texte.

Une `PhraseSem_i` quelconque possède la représentation interne :

```
[GroupeSujet, GroupeVerbal, GroupeInformatif,
GroupeComplément, TextePhrase, TexteVerbe]
```

Le `GroupeSujet` possède la représentation interne :

```
[Mot_1, Mot_2, Roles]
```

où
- `Mot_i` est la liste correspondant aux mots appartenant au groupe sujet (le groupe sujet peut contenir au maximum 2 mots : l'article et le nom) ;
- `Roles` est une liste de `Role` ; et `Role` est une chaîne de caractère représentant le rôle joué par le groupe sujet.

Un `Mot_i` possède la représentation interne

```
[NatureGrammaticale, Canonique, MotRéel, Position]
```

où
- `NatureGrammaticale` est une chaîne de caractères représentant la nature grammaticale du mot ;
- `Canonique` est une chaîne de caractères représentant la forme canonique du mot ;
- `MotReel` est une chaîne de caractères représentant le mot de la phrase;
- `Position` est un entier représentant la position du mot dans la phrase.

Le `GroupeVerbal` possède la représentation interne :

```
[Verbe_1, Verbe_2, SensReel]
```

où
- `Verbe_i` est la liste correspondant aux verbes appartenant au groupe verbal hormis l'auxiliaire être (le groupe verbal ne peut donc contenir au maximum que deux mots : l'auxiliaire de mode et le verbe conjugué) ;

- `SensReel` est le sens réel joué par le groupe verbal.

Un `Verbe_i` possède la représentation interne

```
[NatureGrammaticale, Canonique, MotReel, Substantif,
SensRéel, Position]
```

où
- `NatureGrammaticale` est une chaîne de caractères représentant la nature grammaticale du verbe ;
- `Canonique` est une chaîne de caractères représentant l'infinitif du verbe ;
- `MotReel` est une chaîne de caractères représentant le verbe conjugué ;
- `Substantif` est une chaîne de caractères représentant le substantif du verbe ;
- `Semantique` est une chaîne de caractères représentant le sens du verbe ;
- `Position` est un entier représentant la position du verbe dans la phrase.

Le `GroupeInformatif` possède la représentation interne :

```
[Info_1, Info_n]
```

où `Info_i` est une liste correspondant aux mots du groupe informatif.

Une `Info_i` possède la représentation interne

```
[NatureGrammaticale, Canonique, MotReel, Position, Roles]
```

où
- `NatureGrammaticale` est une chaîne de caractères représentant la nature grammaticale du mot ;
- `Canonique` est une chaîne de caractères représentant la forme canonique du mot ;
- `MotReel` est une chaîne de caractères représentant le mot de la phrase;
- `Position` est un entier représentant la position du mot dans la phrase.

Le `GroupeComplement` possède la représentation interne :

```
[Mot_1, Mot_2, ..., Mot_n, Roles]
```

où
- `Mot_i` est une liste correspondant aux mots appartenant au groupe complément ;
- `Roles` est une liste de `Role` ; et `Role` est une chaîne de caractère représentant le rôle joué par le groupe complément.

Un `Mot_i` possède la représentation interne :

```
[Numero, NatureGrammaticale, Canonique, MotReel, Position]
```

où
- `Numero` est un entier représentant le numéro du groupe complément ;

- `NatureGrammaticale` est une chaîne de caractères représentant la nature grammaticale du mot ;
- `Canonique` est une chaîne de caractères représentant la forme canonique du mot ;
- `MotReel` est une chaîne de caractères représentant le mot de la phrase;
- `Position` est un entier représentant la position du mot dans la phrase.

Le `TextePhrase` est une chaîne de caractères représentant le texte de la phrase et le `TexteVerbe` est une chaîne de caractères représentant le texte du groupe verbal.

Annexe VIII : Lexique de base

Le lexique de base représente la connaissance de notre outil logiciel sur le vocabulaire et la grammaire anglaise. Nous donnons ci-après le lexique de base utilisé par l'étude de cas du chapitre 7. Remarquons que le lexique de base ne contient aucun verbe d'action.

Forme canonique	Type Grammatical	Information (forme primitive)	Sens du mot
A	ARTICLE		NULL
AN	ARTICLE		NULL
AND	CONJUNCTION		NULL
AT-LEAST	CONSTRAINT		MINIMUM
AT-MOST	CONSTRAINT		MAXIMUM
BE	VERB	IS-A	SPECIALISATION
BY	PREPOSITION		NULL
CAN	VERB		AUXILIARY
CHARACTERIZE	VERB		POS-COMP
COMPOSE	VERB	COMPOSITION	POS-COMP
DECOMPOSE	VERB	DECOMPOSITON	POS-COMP
EACH	ARTICLE		NULL
EITHER	CONSTRAINT		OTHER
EVERY	ARTICLE		NULL
HAVE	VERB	HAVING	POS-SUJ
HIS	ARTICLE		NULL
IDENTIFY	VERB	IDENTIFICATION	IDENT-COMP
ITS	ARTICLE		NULL
MAXIMUM	CONSTRAINT		MAXIMUM
MUST	VERB		AUXILIARY
ONLY	CONSTRAINT		MINIMUM
OR	DISJUNCTION		NULL
SEVERAL	CONSTRAINT		MAXIMUM
SOME	ARTICLE		NULL
SPECIFY	VERB	SPECIFICATON	POS-COMP
THE	ARTICLE		NULL
THEIR	ARTICLE		NULL